ESTIMATION DES JONCTIONS, DU MOUVEMENT APPARENT ET DU RELIEF EN VUE D'UNE RECONSTRUCTION 3D DE LA SCÈNE

par

François Deschênes

mémoire présenté au Département de mathématiques et d'informatique
en vue de l'obtention du grade de maître ès sciences (M.Sc.)

FACULTÉ DES SCIENCES
UNIVERSITÉ DE SHERBROOKE

Sherbrooke, Québec, Canada, décembre 1999

Le 4 février 2000 , le jury suivant a accepté ce mémoire dans sa version finale.
date

Président-rapporteur:	M. Jean Goulet Département de mathématiques et d'informatique
Membre:	M. Shengrui Wang Département de mathématiques et d'informatique
Membre:	M. Djemel Ziou Département de mathématiques et d'informatique

SOMMAIRE

La reconstruction 3D d'une scène réelle à partir d'une ou plusieurs images permet de comprendre la relation tridimensionnelle qui existe entre les différents objets. À cet effet, il existe plusieurs indices de profondeur (coins, flou, disparité, mouvement, etc.) permettant la perception 3D. Pour notre part, nous avons étudié, développé et implanté quatre approches dont une pour la détection des jonctions de lignes, une pour la segmentation basée sur le mouvement, une pour l'estimation du flou et enfin une pour l'estimation simultanée du flou et de la disparité. La détection des jonctions et des terminaisons de lignes est basée sur une mesure de courbure locale. Étant donné que ces caractéristiques de l'image sont robustes, elles constituent un intérêt particulier pour la mise en correspondance et la reconstruction 3D. La segmentation en couches des images d'une séquence vidéo consiste à déterminer l'ordre de profondeur relatif des objets (régions) les uns par rapport aux autres à partir d'un critère d'homogénéité du flot optique. L'estimation de la différence de flou entre deux images d'une même scène est fondée sur la transformée d'Hermite. Afin de valider l'estimation résultante, la profondeur des objets de la scène est ensuite calculée à partir de la différence de flou. Finalement, le modèle unifié permet de calculer de manière simultanée et coopérative le flou et la disparité. Il constitue en fait une généralisation de notre approche d'estimation du flou. Les indices ainsi calculés peuvent ensuite servir à la reconstruction 3D de scènes complexes.

REMERCIEMENTS

Je tiens à exprimer en premier lieu toute ma reconnaissance envers mon directeur de maîtrise, le professeur Djemel Ziou. Son encadrement exemplaire, ses conseils avisés et sa disponibilité furent pour moi des éléments importants quant à la bonne conduite de mes projets de recherche.

Je tiens à remercier en second lieu Jean Goulet, professeur et doyen de la Faculté des sciences. Par ses suggestions et ses critiques constructives, il m'a permis d'améliorer mes capacités d'analyse tout en développant mes aptitudes pour l'enseignement.

Mes remerciements vont également à l'ensemble des membres du groupe de recherche en vision et traitement d'images pour leur agréable compagnie et leur soutien scientifique. Plus particulièrement, je tiens à remercier le professeur Shengrui Wang, Marie-Flavie Auclair Fortier, Scott Lawrence et Mohammed Ouali pour leurs précieux conseils tout au long de cette maîtrise. Les échanges avec ces derniers ainsi que leur expertise ont favorisé l'avancement et l'amélioration de mes travaux.

Enfin, je tiens à remercier ma famille et toutes les personnes qui me sont proches pour leurs encouragements soutenus et leur patience. Des remerciements spéciaux sont dédiés à Caroline Canuel, ma compagne de tous les jours, pour le temps qu'elle a consacré à la lecture de mes nombreux travaux et pour ses précieuses suggestions quant à l'amélioration de ces derniers sur le plan de la rédaction.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

INTRODUCTION

La reconstruction tridimensionnelle d'une scène réelle constitue une tâche fondamentale pour de nombreux domaines d'application. Elle permet de comprendre les relations 3D qui existent entre les différents objets de l'espace. Elle rend notamment possible les chirurgies assistées par ordinateur, le développement de systèmes de guidage automatique, la création d'effets spéciaux de haute qualité et le développement de simulateurs réalistes. Elle est donc largement utilisée dans les applications de la vision par ordinateur et du traitement d'images en médecine, en robotique, au cinéma ainsi qu'en aérospatiale. La reconstruction 3D implique généralement l'extraction de caractéristiques pertinentes contenues dans les images et associées à la perception 3D. Ces caractéristiques sont communément appelées des indices de profondeur. Parmi les plus connues, nous retrouvons les jonctions, le mouvement, le flou et la disparité. En ce sens, nous avons étudié, développé et implanté quatre approches pour l'extraction directe ou indirecte des indices de profondeur suivants : jonctions de lignes, mouvement, flou et disparité.

La première approche concerne la détection des jonctions. Ces dernières représentent des informations robustes de l'image. Elles constituent donc un intérêt particulier pour la mise en correspondance et la reconstruction 3D. Cependant, à ce jour, aucun détecteur de coins ne semble avoir été développé pour les intersections et extrémités de lignes. Afin d'extraire ce type de jonctions, nous proposons une approche qui se divise en deux

étapes. La première consiste à calculer la courbure locale des lignes. À cet effet, nous avons développé, testé et comparé deux mesures différentes : le taux de changement de direction des vecteurs d'orientation le long de la ligne et la moyenne des produits scalaires des vecteurs d'orientation à l'intérieur d'un voisinage donné. La seconde étape consiste à localiser les jonctions et terminaisons à partir de la courbure calculée.

La seconde approche est destinée à l'estimation du mouvement apparent présent dans une séquence vidéo en vue de déterminer l'ordre de profondeur relatif des objets les uns par rapport aux autres. À titre d'exemple, lorsque l'on se déplace en voiture, les objets éloignés semblent se déplacer moins rapidement que ceux rapprochés. À cet effet, nous proposons un algorithme qui, dans un premier temps, calcule le flot optique (mouvement apparent). Par la suite, une segmentation est appliquée aux images à partir d'un critère d'homogénéité du flot optique. Les résultats obtenus consistent en des couches (régions) classées selon leur ordre de profondeur relatif.

La troisième approche concerne l'estimation dense et précise de la différence de flou à partir de deux images de la même scène obtenues en faisant varier les paramètres intrinsèques de la caméra (distance focale, ouverture, etc.). Pour ce faire, nous démontrons que l'image la plus floue peut être exprimée en fonction de la différence de flou et des dérivées partielles de la seconde image. Ainsi, la différence de flou peut être calculée en résolvant un système d'équations, ce qui mène à une précision, une densité et une efficacité plus élevées que celles fournies par des algorithmes existants. En guise de validation, l'estimation résultante est ensuite utilisée pour construire une carte de profondeurs dense et précise des scènes réelles.

Finalement, la dernière approche permet de calculer de manière simultanée et coopérative deux indices de profondeur : la disparité et le flou. Il s'agit d'une généralisation de notre approche d'estimation de la profondeur à partir du flou. En ce sens, nous démontrons qu'une image provenant d'une paire stéréoscopique ou d'une séquence d'images peut être

exprimée en fonction des dérivées partielles d'une seconde image, d'une variation de flou et d'un facteur de décalage. Cela suppose donc que ces derniers peuvent être estimés en résolvant un système d'équations. Les indices de profondeur ainsi calculés peuvent ensuite servir à la reconstruction précise de scènes réelles.

Le mémoire est divisé en quatre chapitres. Le premier concerne la détection des jonctions et des extrémités de lignes. Le deuxième chapitre détaille l'estimation du mouvement dans un contexte de segmentation en couches des images. Le chapitre 3 décrit l'approche d'estimation des variations de flou en vue d'estimer la profondeur des points de la scène. Finalement, le dernier chapitre présente le modèle unifié d'estimation du flou et de la disparité pour des images unidimensionnelles.

CHAPITRE 1

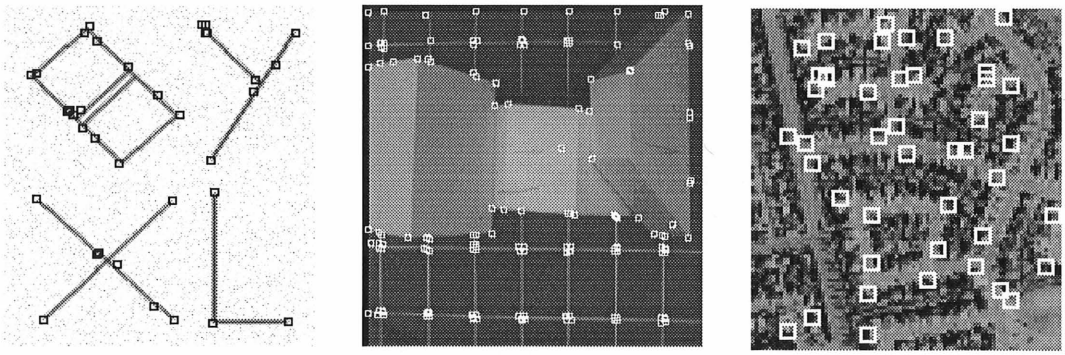
DÉTECTION DES JONCTIONS ET DES EXTRÉMITÉS DE LIGNES

1.1 Introduction

Les contours constituent l'une des caractéristiques de l'image la plus utilisée dans le domaine de la vision par ordinateur. Ils correspondent à des discontinuités ou variations locales des niveaux de gris. Ces dernières sont généralement produites par la projection sur le plan image des contours physiques des objets de la scène ou bien par des changements d'illumination ou de réflectance. Communément, le terme *contour* englobe tous les types de contours, dont les plus connus sont les marches, les vallées et les crêtes. Les marches correspondent habituellement aux frontières entre des régions ayant des niveaux de gris respectifs quasiment constants, mais différents. Pour leur part, les vallées et les crêtes proviennent généralement d'objets fins placés devant un fond, de routes et de rivières dans les images aériennes ou satellitaires, ou de l'illumination mutuelle entre des objets de la scène qui sont en contact. Les contours comportent certains points représen-

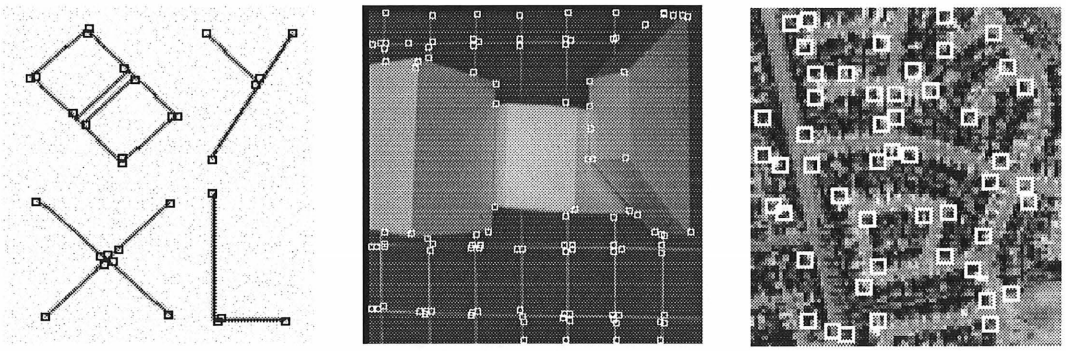
tant des informations particulièrement robustes. Parmi ceux-ci, nous retrouvons les jonctions, c'est-à-dire les intersections de plusieurs contours, communément appelées *coins*. Nous retrouvons également les extrémités, ou terminaisons, lesquelles correspondent aux points terminaux d'un contour. Les coins se sont avérés utiles dans l'accomplissement de nombreuses tâches de la vision par ordinateur et du traitement d'images, telles que l'estimation du flot optique [36], la mise en correspondance stéréoscopique [8] et la détection des intersections de routes dans les images satellitaires [16]. Pour leur part, les terminaisons sont employées pour la détection des extrémités de routes dans les images aériennes à faible résolution [16] et dans les algorithmes de fermeture de contours [27, 33]. La détection des jonctions et des extrémités de contours constitue donc une tâche majeure.

Les détecteurs de coins traditionnels concernent principalement les jonctions de marches [6, 9, 22, 27, 53]. Ces détecteurs utilisent généralement une recherche implicite ou explicite des variations locales des niveaux de gris associées à ce type de contours. Cependant, ces variations diffèrent de celles produisant les crêtes et les vallées [64]. Par conséquent, les résultats fournis par les détecteurs de coins de marches ne sont pas adéquats en présence de jonctions de crêtes ou de vallées. En fait, tel qu'illustré par les figures 1.1 et 1.2, dans une telle situation, leur réponse n'est pas unique et elle est imprécise. Les contours de



a. Synthétiques b. Lignes réelles c. Routes

Figure 1.1 – *Résultats fournis par le détecteur de jonctions de marches de Tabbone*



a. Synthétiques b. Lignes réelles c. Routes

Figure 1.2 – *Résultats fournis par le détecteur de jonctions de marches CSS*

type vallée ou crête sont communément appelés des *lignes*. Ceux-ci sont utilisés dans de nombreuses applications incluant la mise à jour automatique de bases de données cartographiques [16] et l'extraction de caractéristiques de l'anatomie humaine en imagerie médicale [7]. Nous nous intéressons plus particulièrement à l'extraction des jonctions et des extrémités de lignes.

Ce chapitre décrit une méthode de détection des jonctions L, X, Y, et T ainsi que des terminaisons de lignes dans une image à niveaux de gris. Selon la littérature, ce problème ne semble pas avoir été réellement abordé à ce jour. L'approche présentée se divise en deux étapes principales. D'abord, à partir des informations de lignes extraites de l'image originale (position, plausibilité et orientation), la courbure locale est calculée. À cette fin, deux mesures différentes ont été développées, testées et comparées. La première consiste en la projection du taux de changement de direction des vecteurs d'orientation le long de la ligne. La seconde mesure est la moyenne des produits scalaires des vecteurs d'orientation à l'intérieur d'un voisinage donné. Par la suite, les jonctions et terminaisons de lignes sont localisées à partir de la courbure locale.

La section suivante présente un survol des algorithmes de détection de coins de type marche existants. Les sections 1.3 et 1.4 sont dédiées à une description détaillée de l'es-

timation de la courbure et de la localisation des jonctions, respectivement les première et deuxième étapes de la méthode proposée. La section 1.5 concerne la détection des extrémités de lignes. L'algorithme est résumé à la section 1.6. Les résultats expérimentaux ainsi qu'une description de l'influence des différents paramètres sont présentés à la section 1.7.

1.2 Travaux connexes

À notre connaissance, nous proposons le premier détecteur de coins et de terminaisons développé pour les lignes [10]. Puisque ce dernier est inspiré des approches suggérées pour les contours de type marche, cette section constitue un survol rapide des principales techniques proposées pour ce type de contour. Au niveau des détecteurs de jonctions, Beaudet [4] propose un opérateur invariant par rotation, appelé DET, qui correspond au déterminant de la matrice hessienne. Il suggère que la détection des coins peut être effectuée directement en seuillant les valeurs absolues des extremums de cet opérateur. Pour leur part, Kitchen et Rosenfeld [22] présentent quatre opérateurs basés respectivement sur le module du gradient de la direction du gradient, la différence entre les directions du gradient le long du contour à l'intérieur d'un voisinage, l'angle entre les voisins les plus similaires et finalement la courbure d'une surface polynomiale du second ordre approchant des voisinages carrés. Le comportement dans l'espace échelle de ces détecteurs fut étudié par Deriche et Giraudon [9] à l'aide de modèles idéals de jonctions L et T. Ils ont démontré que ces détecteurs ne localisent pas les coins avec précision. Étant donné que l'emplacement exact d'un coin correspond à un passage par zéro du Laplacien et qu'il est insensible à l'échelle, ils suggèrent de combiner le Laplacien et l'opérateur DET afin d'améliorer la précision des résultats. Cependant, il a été démontré que pour plusieurs modèles de coins (e.g. ceux formés par des courbes non linéaires), il n'y a pas de passage

par zéro du Laplacien. L'utilisation des algorithmes qui localisent les coins à l'aide de ce type de passages par zéro est donc limité [54]. Tabbone [53] propose un algorithme qui utilise l'extremum elliptique (maximum ou minimum local) du Laplacien de la gaussienne (LOG). Il a démontré que l'emplacement de cet extremum elliptique est toujours à l'intérieur du coin et que, dans l'espace échelle, l'extremum se déplace sur la bissectrice de ce coin. Deux détecteurs additionnels furent suggérés par Cooper *et al.* [6]. Le premier utilise la dissimilitude entre les régions de l'image dans la direction du contour afin de détecter les coins. Le second estime la courbure à partir de la deuxième dérivée de l'image dans la direction du contour. Pour leur part, Lacroix et Acheroy [23] emploient le produit vectoriel des vecteurs gradients pour détecter les changements locaux d'orientation dans la direction du contour. Finalement, Mokhatarian et Suomela [27] présentent une approche basée sur l'évolution de la courbure dans l'espace échelle. À partir des contours extraits à l'aide du détecteur de Canny, ils calculent la courbure à haute échelle et localisent les coins potentiels au maxima de la courbure. Au fur et à mesure que l'échelle décroît, ils effectuent un suivi des coins afin d'améliorer la précision.

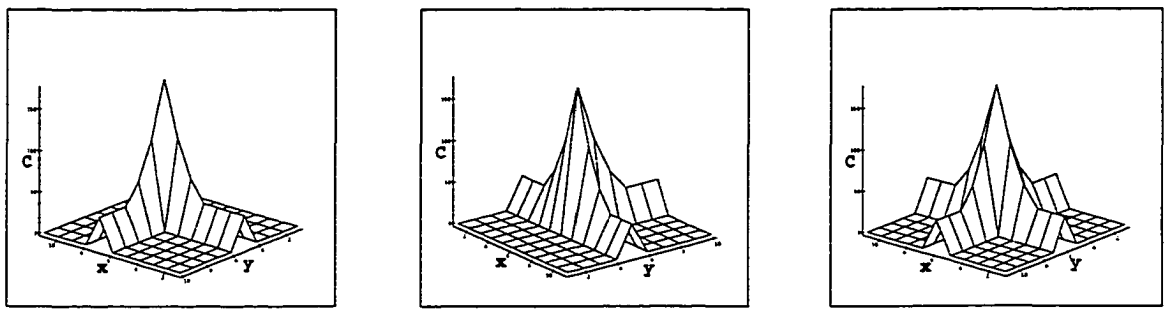
Au niveau des terminaisons, Nitzberg *et al.* [33] définissent une terminaison en tant que jonction T faible. Cela signifie qu'une terminaison correspond en fait à une jonction T formée par l'intersection de deux contours dont l'un est très court. Ils suggèrent ainsi que les extrémités des contours peuvent être directement détectées par les détecteurs de coins existants. Par conséquent, ils ne proposent aucune méthode d'extraction explicite.

Toutes ces approches concernent la détection des coins de marches, mais aucune d'entre elles ne solutionne les problèmes reliés à l'extraction des jonctions et des extrémités de lignes. Les intersections des contours de type marche sont généralement localisées à partir d'un opérateur Laplacien (passages par zéro ou extremums) ou par une mesure de courbure basée sur la direction du gradient. Cependant, le gradient d'une ligne est faible, voire même nul. De plus, une jonction de lignes ne correspond pas à un passage par zéro du

Laplacien. Par conséquent, les réponses des détecteurs de jonctions de marches existants sont imprécises en présence d'une jonction ou d'une terminaison de lignes, c'est-à-dire que plusieurs points sont détectés (figures 1.1 et 1.2).

1.3 Estimation de la courbure locale d'une ligne

Le taux de changement de la direction d'une ligne est communément appelée *courbure*. Selon cette définition, une jonction idéale correspond à un maximum local de la courbure de la ligne, tel qu'illustré par la figure 1.3. En supposant que les lignes (plausibilité,



a. Jonction L b. Jonction T c. Jonction X

Figure 1.3 – *Exemples de courbures théoriques des jonctions*

orientation et position) ont été préalablement extraites de l'image originale à l'aide d'un détecteur de lignes, la courbure locale peut alors être estimée. À cette fin, nous avons développé, testé et comparé deux mesures différentes. La première est inspirée d'une mesure proposée par Kitchen et Rosenfeld [22] pour les coins de marches. La version originale est définie comme étant le taux de changement de la direction du gradient le long du contour multiplié par l'amplitude du gradient. Étant donné que l'amplitude du gradient correspond à la plausibilité des contours de type marche, la valeur de la courbure résultante est élevée pour tout point (x,y) appartenant à un contour de fort contraste qui

tourne rapidement. Comme le gradient d'une ligne est faible, voire même nul [66], cette mesure n'est pas adaptée aux lignes. Elle doit donc être modifiée. Pour ce faire, calculons le taux de changement de direction à partir de $\vec{d}(x,y) = (u,v)$, le vecteur d'orientation non normalisé perpendiculaire à un point (x,y) de la ligne. La direction θ de $\vec{d}$ est donnée par :

$$\theta(x,y) = \begin{cases} \arctan\left(\frac{v}{u}\right) & \text{si } u \neq 0 \\ \frac{\pi}{2} & \text{sinon.} \end{cases} \tag{1.1}$$

Les dérivées partielles de θ sont respectivement :

$$\theta_x = \frac{v_x u - u_x v}{u^2 + v^2} \qquad \text{et} \qquad \theta_y = \frac{v_y u - u_y v}{u^2 + v^2}, \tag{1.2}$$

où u_x et v_x représentent les dérivées partielles de u et v. Ainsi, la projection du vecteur (θ_x,θ_y) le long de la ligne, multipliée par l'amplitude de $\vec{d}$, fournit une mesure de courbure donnée par :

$$\begin{aligned} C(x,y) &= \left((\theta_x,\theta_y) \cdot \frac{(-v,u)}{\sqrt{u^2+v^2}}\right)\sqrt{u^2+v^2} \\ &= \frac{u_x v^2 + v_y u^2 - uv(u_y + v_x)}{u^2+v^2}, \end{aligned} \tag{1.3}$$

où $\cdot$ représente le produit scalaire. Cette quantité est grande pour tout point qui appartient à une région avoisinant une jonction (figures 1.5d et 1.6d). Ceci est dû au changement significatif de direction de θ à l'intérieur d'une telle région.

La seconde mesure repose sur la dissimilitude entre les vecteurs d'orientation des pixels de la ligne à l'intérieur d'un voisinage donné. Lacroix et Acheroy [23] ont proposé une mesure de dissimilitude pour les coins de marches basée sur le produit vectoriel des vecteurs gradients. Pour des raisons de commodité, remplaçons le produit vectoriel par un produit scalaire. Ainsi, en adaptant le tout aux lignes, la mesure de dissimilitude entre n'importe quelle paire de vecteurs d'orientation $(\vec{d}(x_1,y_1),\vec{d}(x_2,y_2))$ est fournie par :

$$s(\vec{d}(x_1,y_1),\vec{d}(x_2,y_2)) = 1 - \left|\frac{\vec{d}(x_1,y_1)\cdot\vec{d}(x_2,y_2)}{\|\vec{d}(x_1,y_1)\|\|\vec{d}(x_2,y_2)\|}\right|, \tag{1.4}$$

où $\cdot$ représente le produit scalaire. $s(\vec{d}(x_1,y_1),\vec{d}(x_2,y_2))$ vaut 1 pour des vecteurs perpendiculaires et 0 pour des vecteurs parallèles. La deuxième mesure de courbure peut donc être estimée en calculant la moyenne des $s(\vec{d}(x,y),\vec{d}(x+\Delta x,y+\Delta y))$ au sein d'un voisinage local $(2M+1)\times(2M+1)$. Elle est donnée par :

$$C(x,y) = \frac{1}{N}\sum_{\Delta x=-M}^{M}\sum_{\Delta y=-M}^{M}\beta(\Delta x,\Delta y)\times s(\vec{d}(x,y),\vec{d}(x+\Delta x,y+\Delta y)), \qquad (1.5)$$

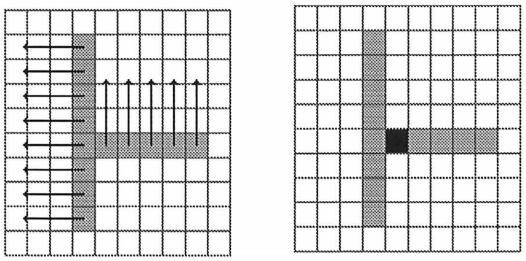
où

$$\beta(\Delta x,\Delta y) = \begin{cases} e^{-(\Delta x^2+\Delta y^2)} & \text{si un chemin formé de pixels ayant une plausibilité non nulle existe entre } (x,y) \text{ et } (x+\Delta x,y+\Delta y) \\ 0 & \text{sinon} \end{cases}$$

et N représente le nombre de points $(x+\Delta x,y+\Delta y)$ tels que $\beta \neq 0$. β garantit que l'influence des voisins éloignés de (x,y) sur C est moindre que celle des voisins rapprochés. Cette fonction assure également que seuls les points appartenant à une ligne sont considérés, ce qui signifie qu'il n'est pas nécessaire d'utiliser explicitement la plausibilité de la ligne dans l'équation (1.5). Ainsi, la valeur de $C(x,y)$ est grande pour tout point de la ligne appartenant au voisinage d'une jonction (figures 1.5e et 1.6e).

1.4 Localisation des jonctions de lignes

On s'attend à ce que les maxima locaux de la courbure correspondent aux jonctions de lignes. Cependant, les valeurs de courbure obtenues à l'aide des mesures présentées à la section précédente indiquent que cette propriété est parfois fausse pour certains types de jonctions, comme le montre la figure 1.4. De plus, le lissage et l'utilisation de dérivées numériques affectent l'estimation de la courbure dans le voisinage d'une jonction. Ils influencent également l'emplacement et l'orientation des lignes. Par conséquent, rien ne garantit que les maxima locaux de courbure coïncident avec les emplacements des

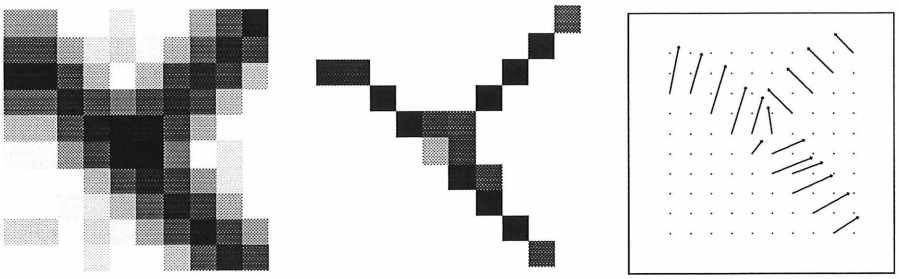


a. Orientations b. Maximum (noir)

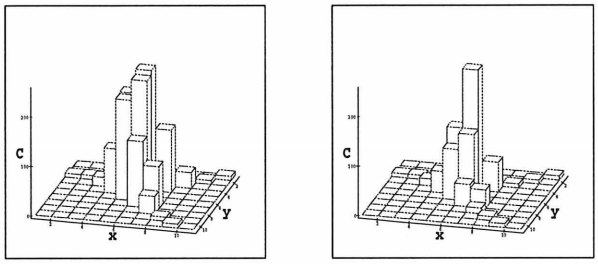
Figure 1.4 – *Maximum local de la courbure d'une jonction T discrète*

jonctions, tel qu'illustré par les figures 1.5 et 1.6. L'algorithme proposé implique donc un traitement additionnel dont l'objectif est la correction de la courbure mesurée.

Par définition, une jonction de lignes est formée par l'intersection de plusieurs lignes dont les orientations respectives sont différentes, ce qui signifie que l'orientation d'un point de jonction n'est pas unique. Toutefois, tel que montré par la figure 1.4a, un détecteur de lignes ne fournit qu'un seul vecteur d'orientation par pixel, et ce, même aux points de jonctions. Cela constitue une source d'imprécision au niveau du calcul de la courbure. Afin de contourner ce problème, extrapolons les vecteurs d'orientation des jonctions à partir de ceux appartenant à leur entourage. Puisque le lissage modifie les vecteurs d'orientation dans les régions de forte courbure (figure 1.5c), il importe de ne pas considérer l'orientation des voisins immédiats des jonctions. À cette fin, deux classes de points de lignes sont créées : ceux ayant une courbure faible (en deçà d'un seuil t_c) et ceux ayant une courbure élevée (supérieure à t_c). Définissons les frontières de faible courbure comme étant les points de lignes ayant une faible courbure et appartenant au voisinage immédiat d'un point de forte courbure (voir la figure 1.7). En se basant sur cette définition, nous déterminons les frontières de faible courbure en vue de prédire l'emplacement des jonctions. Pour une ligne donnée, puisque sa largeur est supposée être égale à un pixel, seulement un point de frontières est considéré à l'intérieur d'un voisinage local. Tous ceux qui ne correspondent pas à des maxima de continuité $p(x,y)$ sont éliminés, où $p(x,y)$ est



a. Jonction lissée b. Lignes extraites c. Orientations



d. Courbure éq. (1.3) e. Courbure éq. (1.5)

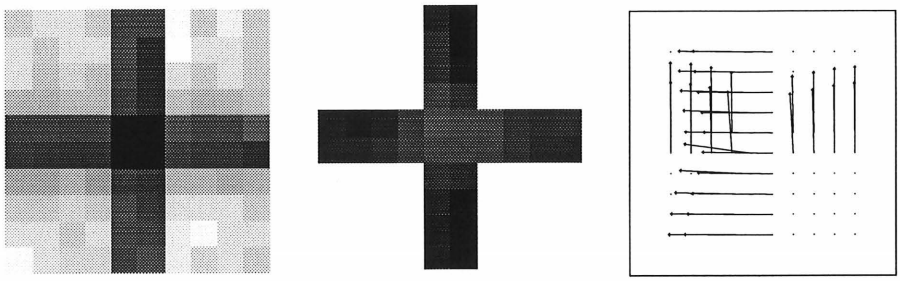
Figure 1.5 – *Exemple d'influence du lissage sur l'estimation de la courbure*

donnée par :

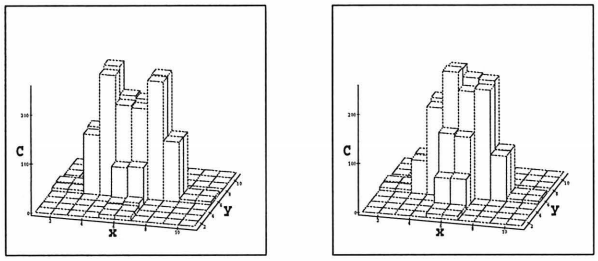
$$p(x,y) = [1 - C(x,y)] \times L(x,y), \tag{1.6}$$

$C(x,y)$ représente la courbure calculée et $L(x,y)$ la plausibilité de la ligne. $p(x,y)$ est maximale pour les points de frontières dont la courbure est la plus faible et la plausibilité la plus élevée. Mentionnons que la plausibilité a déjà été considérée explicitement dans le cas de l'équation (1.3), mais le fait de la considérer une seconde fois n'influence pas le résultat final.

À l'aide des frontières de faible courbure, les vecteurs d'orientation de chacune des jonctions sont ensuite déduits, puis utilisés afin de mettre à jour la courbure. À cette fin, à partir de chacune des frontières, tous les voisins de forte courbure dans la direction de la ligne sont visités à tour de rôle. Un pixel (x,y) de forte courbure est ainsi visité zéro, une ou plusieurs fois. À chaque visite, il reçoit le vecteur d'orientation du pixel frontière



a. Jonction X b. Lignes extraites c. Orientations



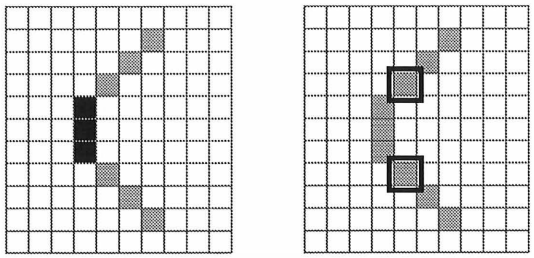
d. Courbure éq. (1.3) e. Courbure éq. (1.5)

Figure 1.6 – *Exemple d'influence des dérivées numériques*

courant ($\vec{d_i}(x,y)$), tel que montré aux figures 1.8b et 1.8c. À la fin des visites, la courbure $C(x,y)$ des pixels de forte courbure est mise à jour selon la règle :

$$C(x,y) = C_o(x,y) + \sum_{i=1}^{n} s(\vec{d_1}(x,y),\vec{d_i}(x,y)) + k, \tag{1.7}$$

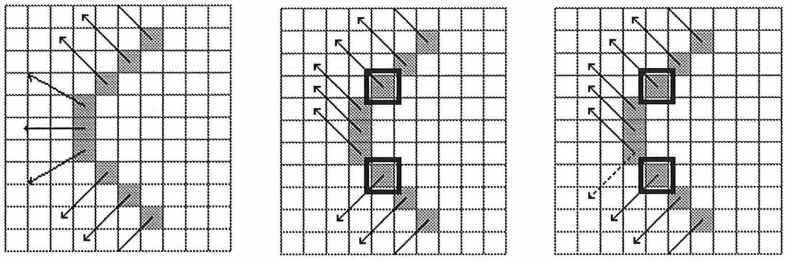
où n représente le nombre de vecteurs d'orientation $\vec{d_i}(x,y)$ associés au pixel (x,y), $C_o(x,y)$ l'estimation originale de la courbure et $s(\vec{d_1}(x,y),\vec{d_i}(x,y))$ est défini par l'équation (1.4). La constante $k \geq 1$ a pour unique effet de distinguer les points n'ayant jamais été visités de ceux l'ayant été au moins une fois. Étant donné que les vecteurs d'orientation des frontières de faible courbure sont généralement similaires à ceux de leurs voisins, ils sont peu influencés par le lissage. Une telle mise à jour garantit donc que les valeurs maximales de la courbure correspondent aux jonctions de lignes. La figure 1.9 présente un exemple des courbures résultantes obtenues à partir des figures 1.5 et 1.6. Conséquem-



a. Forte courbure (pixels noirs)

b. Frontières (pixels encadrés)

Figure 1.7 – *Exemple de frontières de faible courbure*



a. Vecteurs originaux

b. Propagation du 1^{er} vecteur

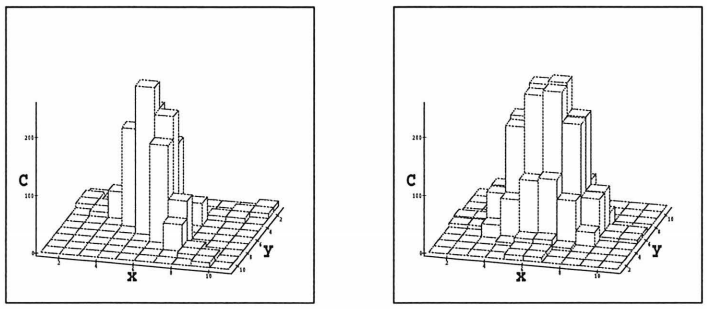
c. Propagation du 2^{e} vecteur

Figure 1.8 – *Propagation des vecteurs d'orientation à partir des frontières*

ment, la localisation des jonctions peut être directement effectuée par l'extraction des points correspondant aux maxima de la courbure.

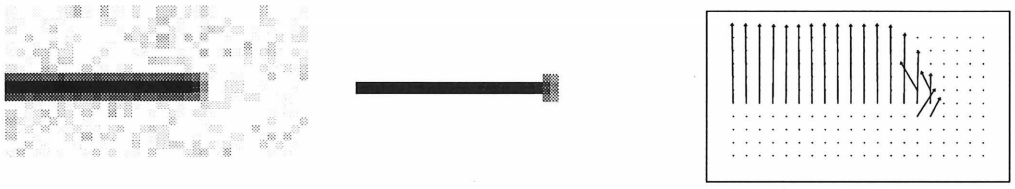
1.5 Méthode de détection des extrémités de lignes

Rappelons qu'une terminaison, telle que définie par Nitzberg *et al.* [33], correspond à une jonction T formée par l'intersection de deux contours dont l'un est très court. Comme le montre la figure 1.10, la réponse d'un détecteur de lignes en présence d'une terminaison confirme cette définition. Dans cette figure, la terminaison est celle de droite. Notons que la plausibilité de la ligne est légèrement plus faible, mais également plus large, à l'empla-



a. Pour figure 1.5 b. Pour figure 1.6

Figure 1.9 – *Courbure résultante*

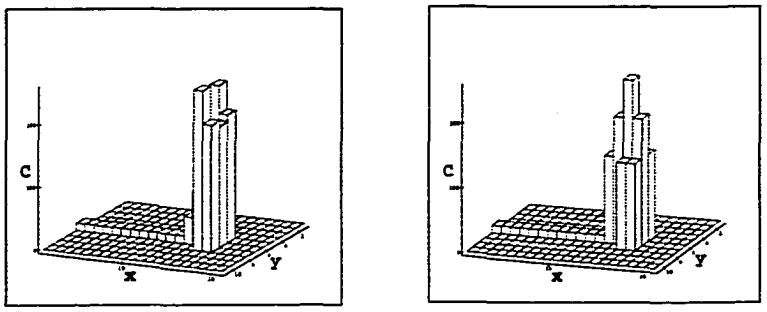


a. Extrémité (droite) b. Ligne extraite c. Orientations estimées

Figure 1.10 – *Réponse d'un détecteur de lignes en présence d'une terminaison*

cement de l'extrémité (figure 1.10b). Ceci est dû au lissage utilisé lors de la détection des lignes. Toutefois, un changement d'orientation est détecté (figure 1.10c). En se basant sur ces propriétés, les terminaisons de lignes peuvent être localisées par la méthode décrite aux sections 1.3 et 1.4. Premièrement, le changement de direction des vecteurs d'orientation à l'intérieur du voisinage local de l'extrémité garantit que la courbure est grande en ce point (équations (1.3) et (1.5)). La figure 1.11a en donne d'ailleurs un exemple. Deuxièmement, à une terminaison donnée est associée une seule frontière de faible courbure. Ainsi, les vecteurs d'orientation sont propagés dans une direction unique qui est celle de la ligne. Cela garantit que l'extrémité en question sera visitée et que sa courbure sera mise à jour. En fait, la présence de la constante d'augmentation minimale k dans l'équation (1.7) assure que tout pixel (x,y) ayant été visité au moins une fois se voit attribuer une valeur de courbure supérieure. Par conséquent, suite à l'étape de mise à

jour, la courbure est maximale au niveau des points de terminaisons (figure 1.11b).



a. Estimation initiale b. Après mise à jour

Figure 1.11 – *Exemple de courbure d'une terminaison de ligne*

1.6 Algorithme résultant

L'algorithme découlant de l'approche proposée se divise en deux étapes principales :

1. Estimation de la courbure locale des lignes à l'aide de l'équation (1.3) ou (1.5). Nous supposons que la plausibilité, l'orientation et la position des lignes ont été préalablement extraites par un détecteur de lignes.
2. Localisation des jonctions et des terminaisons de lignes : à partir de l'estimation de la courbure (C) générée à l'étape 1, un certain nombre de traitements additionnels sont nécessaires afin de localiser les jonctions et terminaisons :

   - Extraction des frontières de faible courbure : les points de lignes ayant une faible courbure et appartenant au voisinage d'un point de forte courbure sont extraits (section 1.4).
   - Mise à jour de la courbure à l'aide de l'équation (1.7).

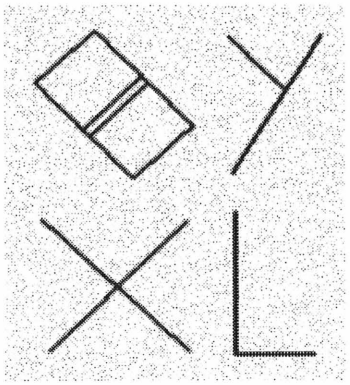
- Extraction des maxima locaux : suite à la mise à jour, les maxima locaux de la courbure coïncident avec les jonctions et terminaisons de lignes. Les points ayant la valeur maximale de courbure à l'intérieur d'un voisinage donné sont donc extraits.

1.7 Évaluation des performances

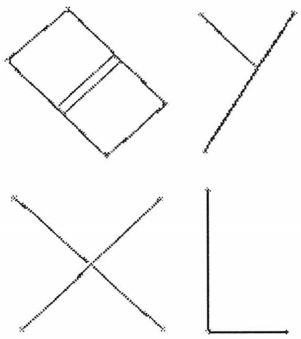
1.7.1 Résultats expérimentaux

La méthode de détection des jonctions et des terminaisons de lignes décrite dans ce chapitre a été testée sur plusieurs images synthétiques et réelles. Pour ce faire, les lignes ont d'abord été extraites des images originales à l'aide d'un algorithme développé par Steger [49]. Au cours des différents tests, il a été prouvé que cet algorithme s'adapte aux lignes de différentes largeurs et qu'il détermine avec une précision subpixel l'emplacement de ces dernières. Cependant, notons qu'il ne constitue pas le coeur de notre approche et qu'il peut être remplacé par tout autre détecteur de lignes fournissant les informations de position, de plausibilité et d'orientation [64].

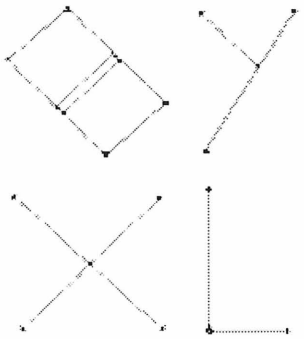
Les résultats suivants ont été obtenus avec quatre images à niveaux de gris. Premièrement, une image synthétique comprenant des jonctions L, T, Y, X et des terminaisons fut utilisée. De manière à la rendre plus réaliste, elle fut d'abord lissée à l'aide d'un filtre gaussien (écart-type = 1.0), puis on lui ajouta un bruit blanc gaussien (écart-type = 6.0). La figure 1.12 présente les résultats obtenus en incluant les étapes intermédiaires. Les paramètres du détecteur de lignes sont l'échelle $\sigma = 1.25$ et le seuil de plausibilité $t = 30.0$. Les valeurs de ces derniers sont sélectionnées de manière à réduire le bruit et éliminer les lignes de faible plausibilité avant de calculer la courbure. Les courbures estimées sont présentées dans les figures 1.12c (équation (1.3)) et 1.12d (équation (1.5)). Seulement les



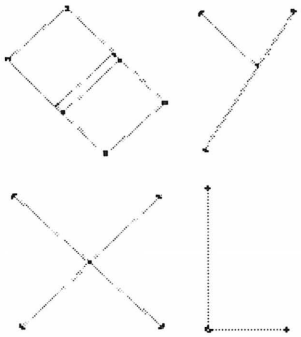
a. Image originale



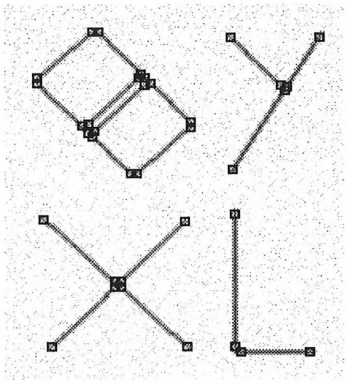
b. Lignes extraites



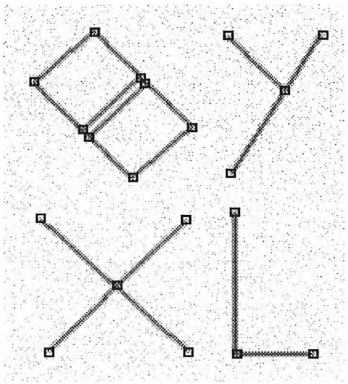
c. Forte courbure éq. (1.3) (zones noires)



d. Forte courbure éq. (1.5) (zones noires)



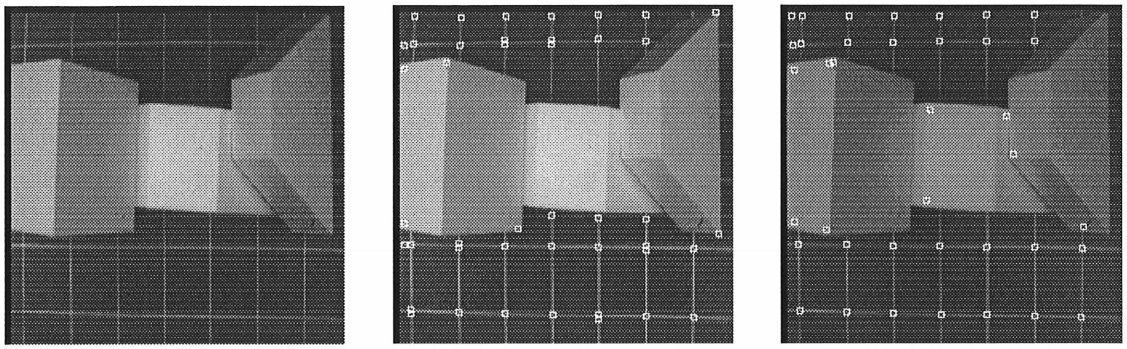
e. Frontières de faible courbure



f. Jonctions et extrémités (éq. (1.3) ou éq. (1.5))

Figure 1.12 – *Image synthétique comprenant des jonctions L, T, Y et X*

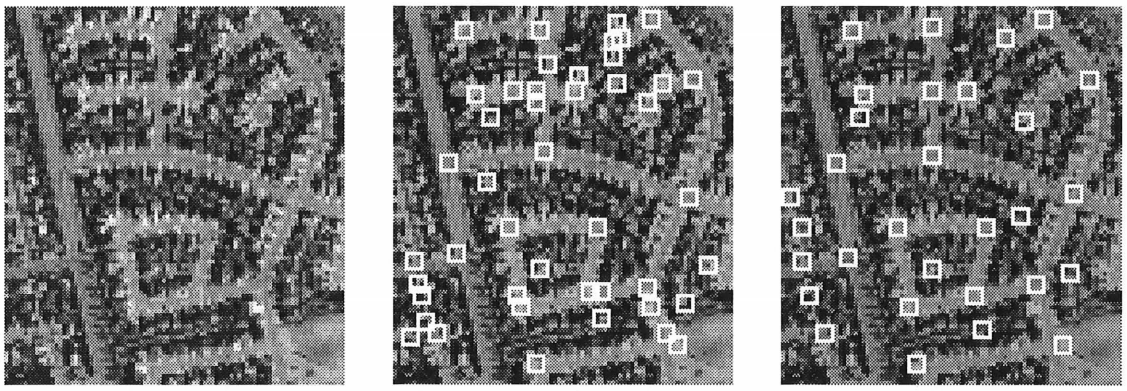
régions de fortes courbure (seuil $t_c = 0.5$ pour l'équation (1.3) et seuil $t_c = 0.9$ pour l'équation (1.5)) sont présentées (régions noires). La seconde courbure (équation (1.5)) a été estimée sur un voisinage 3×3 ($M = 1$). Tel qu'illustré par la figure 1.12f, toutes les jonctions et terminaisons de lignes ont été localisées avec précision, que ce soit avec l'équation (1.3) ou (1.5). Pour sa part, la figure 1.13 présente les résultats découlant d'une image réelle contenant des objets polyédriques placés devant une grille. Les paramètres du détecteur de lignes sont respectivement $\sigma = 2.0$ et $t = 30.0$, tandis que ceux



a. Image originale b. Jonctions (éq. (1.3)) c. Jonctions (éq. (1.5))

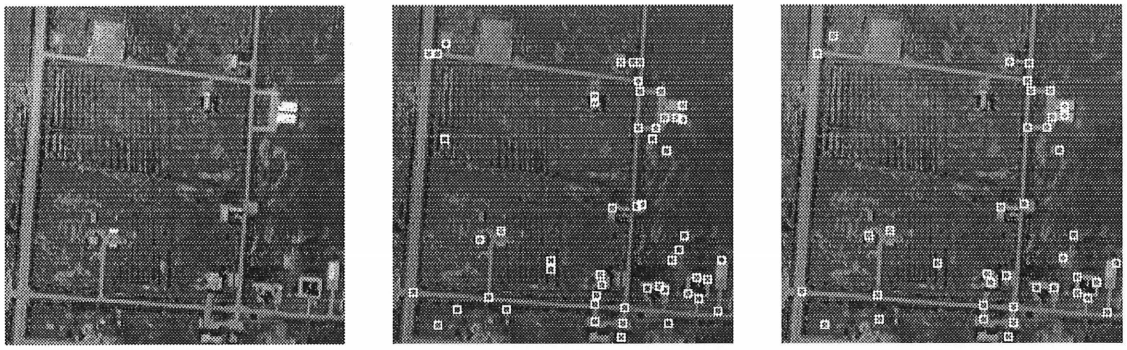
Figure 1.13 – *Image réelle contenant des lignes*

du détecteur de jonctions sont $t_c = 0.31$, (équation (1.3)), et $M = 2$, $t_c = 0.8$ (équation (1.5)). Nous constatons que toutes les jonctions à l'intérieur de la grille ont été détectées. Cependant, notons que pour une jonction donnée, l'équation (1.3) mène parfois à une réponse qui n'est pas unique. Les résultats obtenus à partir d'une image satellitaire contenant des culs-de-sac et des intersections de routes sont présentés dans la figure 1.14 (détecteur de lignes : $\sigma = 2.2$, $t = 30.0$; équation (1.3) : $t_c = 0.2$; équation (1.5) : $M = 2$, $t_c = 0.8$). Mentionnons que l'image originale 1.14a provient de CTI Ottawa, Géomatique Canada. La figure 1.15 présente les résultats provenant



a. Image originale b. Jonctions (éq. (1.3)) c. Jonctions (éq. (1.5))

Figure 1.14 – *Image réelle comprenant des intersections de routes et des culs-de-sac*



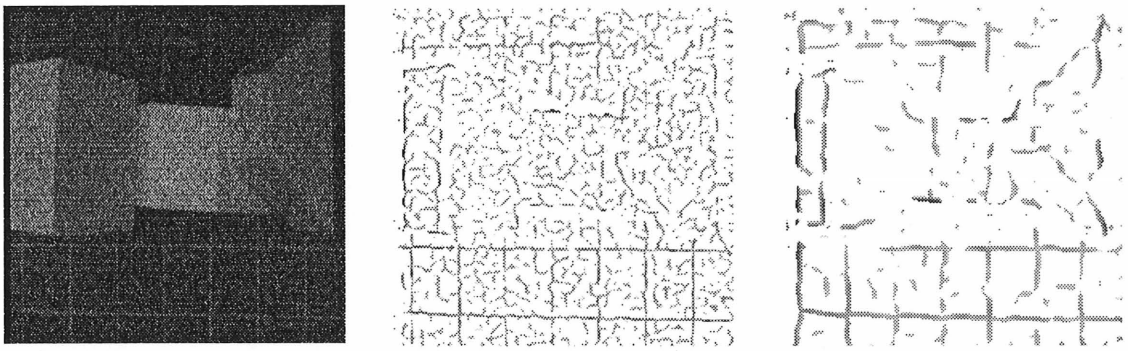
a. Image originale b. Jonctions (éq. (1.3)) c. Jonctions (éq. (1.5))

Figure 1.15 – *Seconde image de routes*

d'une photo aérienne de routes (détecteur de lignes : $\sigma = 2.5$, $t = 55.0$; équation (1.3) : $t_c = 0.2$; équation (1.5) : $M = 2$ et $t_c = 0.4$). Comme le montrent ces deux figures, toutes les extrémités et intersections de routes évidentes ont été localisées. Notons une fois de plus que la précision des résultats fournis par l'équation (1.5) semble être supérieure à celle fournie par l'équation (1.3). En effet, en présence d'une jonction donnée, la réponse de l'équation (1.5) est unique et la localisation est plus précise. Cette différence est principalement due à l'imprécision des dérivées numériques qui sont utilisées dans l'équation (1.3). De plus, $\beta(\Delta x,\Delta y)$ garantit que seulement les points de lignes contigus sont considérés dans l'équation (1.5). Ces résultats confirment que la méthode de détection proposée permet de localiser avec précision les jonctions et terminaisons de lignes. Ils montrent également la supériorité de l'équation (1.5) par rapport à l'équation (1.3). Par ailleurs, mentionnons que cette méthode a aussi été validée dans une application de mise à jour automatique de bases de données cartographiques [16].

1.7.2 Influence des paramètres

Examinons maintenant l'effet des différents paramètres sur la qualité des résultats obtenus. Tel que mentionné précédemment, σ et t représentent respectivement l'échelle et le seuil de plausibilité des lignes. Comme le montre la figure 1.16, l'augmentation de σ réduit le bruit. Cependant, les expérimentations ont montré que cela fait également



a. Image bruitée b. $\sigma = 2.0$ c. $\sigma = 4.0$

Figure 1.16 – *Extraction de lignes à partir d'une image bruitée*

disparaître certaines lignes réelles, ce qui peut signifier une diminution de la plausibilité des lignes. Ainsi, lorsque σ augmente, la valeur de t doit diminuer, et inversement. Le lissage influence également les vecteurs d'orientation. Lorsque de grandes valeurs de σ sont employées, les variations au niveau de la direction de ces vecteurs sont atténuées. La valeur de la courbure locale décroît donc au fur et à mesure que σ augmente. Cette valeur est également influencée par la taille du voisinage (M). La figure 1.17 montre la courbure moyenne C en fonction de M calculée pour l'image de synthèse de la figure 1.12. Les lignes pointillées représentent un intervalle de confiance correspondant à $C\pm$ un écart-type. Comme le montre ce graphique, C tend à diminuer aux points de jonctions lorsque M augmente. Par conséquent, le seuil de courbure t_c doit être ajusté selon les valeurs de σ et de M. Ainsi, lorsque σ et/ou M augmentent, le seuil de courbure t_c doit être diminué afin d'éviter d'éliminer des jonctions ou des terminaisons réelles.

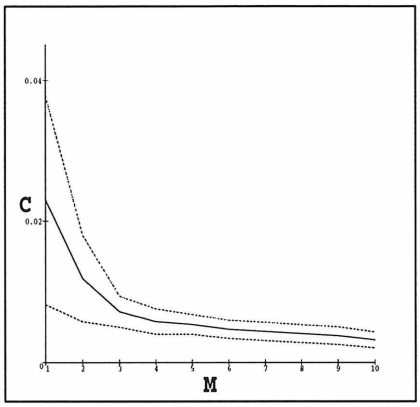
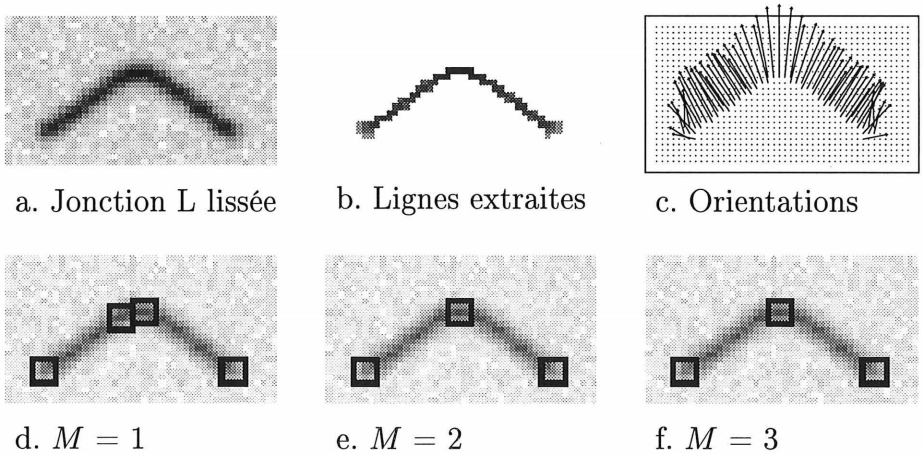


Figure 1.17 – *Influence de M sur le calcul de courbure C (équation (1.5))*

Concernant le choix des paramètres σ et M, il n'existe aucune règle précise. Une petite valeur de M, respectivement σ, peut rendre le détecteur sensible au bruit. Cela signifie qu'en présence d'une jonction unique, de faibles variations d'orientation peuvent occasionner des réponses multiples. L'utilisation d'un grand M et/ou d'un grand σ peut permettre de contourner ce problème. La figure 1.18 présente un exemple d'une telle situation. Cependant, des expérimentations ont montré que cela peut également occasionner



a. Jonction L lissée b. Lignes extraites c. Orientations

d. $M = 1$ e. $M = 2$ f. $M = 3$

Figure 1.18 – *Influence de M (équation (1.5)) sur la localisation des jonctions*

la fusion de certaines jonctions contiguës. C'est-à-dire qu'en présence de deux jonctions rapprochées, la détection de l'une peut-être influencée par la présence de l'autre. Dans un tel cas, rien ne garantit que le maximum local de courbure d'une ligne correspond à l'emplacement d'une jonction. Cela occasionne donc un problème de délocalisation. En conclusion, les valeurs de σ et M doivent assurer un compromis entre les réponses multiples et la fusion des jonctions voisines.

1.8 Conclusion

Les jonctions et terminaisons de lignes diffèrent de celles des contours de type marche. La réponse des détecteurs de coins de marches en présence d'une jonction de lignes n'est pas unique, comme le montrent les figures 1.1 et 1.2. La détection de ce type d'éléments est un problème qui ne semble pas avoir été réellement abordé à ce jour. Nous avons donc proposé une approche pour la détection des jonctions L, X, Y, et T ainsi que des terminaisons de lignes dans une image à niveaux de gris. Celle-ci se divise en deux étapes principales. D'abord, à partir des lignes extraites de l'image originale, la courbure locale est calculée. À cet effet, nous avons développé, testé et comparé deux mesures : le taux de changement de direction des vecteurs d'orientation le long de la ligne (équation (1.3)) et la moyenne des produits scalaires des vecteurs d'orientation à l'intérieur d'un voisinage donné (équation (1.5)). Ensuite, un traitement additionnel, basé sur la courbure estimée et les informations du voisinage, est effectué en vue de localiser avec précision les jonctions et extrémités.

Une évaluation subjective de l'algorithme a été effectuée pour chacune des deux mesures de courbure proposées. En ce sens, nous avons constaté que la seconde mesure permet d'obtenir une réponse unique à une jonction de lignes. De plus, les jonctions et terminaisons évidentes sont localisées. Mentionnons également que ce détecteur a été évalué dans

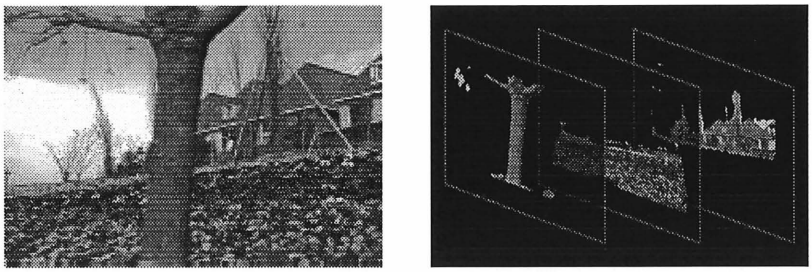
une application réelle. En fait, il a été intégré puis validé dans un système traitant les images satellitaires de réseaux routiers [16]. Cependant, des améliorations restent à faire, notamment l'obtention d'une précision subpixel et l'utilisation du multi-échelles.

CHAPITRE 2

ESTIMATION DU MOUVEMENT DANS UN CONTEXTE DE SEGMENTATION D'IMAGES

2.1 Introduction

La segmentation en couches se définit comme étant la partition (division) d'une ou plusieurs images en régions de profondeurs différentes. Une couche correspond à un ensemble de pixels connexes répondant à un critère d'homogénéité de profondeur. Par définition, une partition sous-entend que chaque pixel de l'image doit appartenir à une et une seule couche. C'est-à-dire que l'intersection de deux couches quelconques est vide et que l'union de toutes les couches redonne l'image originale. Puisqu'il peut exister plusieurs solutions, la partition qui maximise une mesure de qualité est généralement retenue. La figure 2.1 illustre ce concept. Ce type de segmentation peut être utilisé dans plusieurs applications comme la compression de séquences d'images, le suivi d'objets en mouvement et la



a. Image originale b. Image segmentée

Figure 2.1 – *Segmentation d'une image en couches*

création d'effets spéciaux au cinéma grâce à l'insertion ou le retrait de couches.

Supposons que le mouvement présent dans une séquence d'images (mouvement 2D) provient d'une caméra subissant une translation perpendiculaire à l'axe optique au sein d'une scène statique. Notons qu'une telle supposition se justifie par l'application développée. Il a été démontré que la segmentation en couches des images basée sur un critère d'homogénéité de ce type de mouvement permet de déterminer l'ordre de profondeur des différentes couches [58]. En d'autres termes, les régions formées de pixels se déplaçant dans la même direction et ayant une vitesse similaire correspondent généralement à une ou plusieurs surfaces (objets) se trouvant à une même distance de l'observateur. À titre d'exemple, lorsque nous nous déplaçons en voiture, nous remarquons que les objets éloignés se déplacent moins rapidement que ceux rapprochés. Il est évident que la précision des résultats de la segmentation est directement reliée à la précision de l'estimation du mouvement 2D. La méthode d'estimation de ce dernier a donc une importance primordiale.

Le présent chapitre concerne l'estimation du mouvement présent dans une séquence d'images en vue d'effectuer une segmentation en couches. Plus précisément, nous effectuerons dans un premier temps une évaluation comparative des principaux algorithmes d'estimation du mouvement 2D basés sur le flot optique, c'est-à-dire le mouvement ap-

parent de l'intensité lumineuse. Contrairement aux travaux d'évaluation existants [3], les différents algorithmes ne seront pas comparés dans un cadre général, mais plutôt selon des critères découlant du contexte de la segmentation d'images. Dans un deuxième temps, nous détaillerons un algorithme de segmentation en couches simple et rapide basé sur un critère d'homogénéité du flot optique.

La section suivante donne un aperçu des approches de segmentation en couches existantes. La section 2.3 décrit de manière détaillée le flot optique et les problèmes afférents. Un examen rapide des principaux algorithmes d'estimation du mouvement est présenté à la section 2.4. La comparaison de ces algorithmes dans un contexte de segmentation d'images est ensuite établie à la section 2.5. La section 2.6 détaille l'approche de segmentation suggérée. Les résultats expérimentaux ainsi qu'une description de l'influence des différents paramètres sont présentés à la section 2.7.

2.2 Travaux connexes

Diverses méthodes de segmentation en couches ont été suggérées au cours des dernières années. Pour leur part, Murray et Buxton [28] proposent une approche Bayesienne qui effectue une estimation et une segmentation du flot optique simultanément. Celle-ci se base sur une recherche de la probabilité maximale *a posteriori* d'une interprétation (ou segmentation) des données de flot optique. Elle implique la spécification *a priori* des attentes au niveau du mouvement apparent, notamment le nombre de régions devant être extraites, ainsi que l'établissement d'une mesure de qualité de la prédiction du flot optique. La qualité des résultats dépend directement de ces deux éléments. De plus, selon Murray et Buxton, la complexité algorithmique de cette approche est très élevée. Afin de contourner ces problèmes, Wang et Adelson [58] suggèrent une méthode de segmentation basée sur la fusion des pixels dont les propriétés affines du mouvement sont semblables. À

cette fin, une estimation dense du mouvement apparent est d'abord calculée. Cette estimation est ensuite divisée en blocs rectangulaires tels que leur intersection est vide. Pour chacun d'eux, les propriétés affines du mouvement sont estimées. Ces propriétés servent ensuite à effectuer le regroupement des blocs en un nombre donné, généralement petit, de classes à l'aide de l'algorithme des k-moyennes. Finalement, chacun des vecteurs du flot optique est affecté à l'une des classes résultantes, représentées par leur centre, à l'aide d'un classificateur basé sur l'erreur résiduelle minimale. Selon les auteurs, de manière générale, cet algorithme fournit des résultats adéquats. Cependant, certains chercheurs, tels que Borshukov *et al.* [5], ont montré à l'aide d'expérimentations que l'utilisation de l'algorithme des k-moyennes est à la source de certaines imprécisions. D'une part, ils suggèrent d'éviter l'utilisation de cet algorithme de sorte que chaque couche ne soit plus créée à partir d'une simple moyenne des propriétés affines du modèle du mouvement. Ils proposent plutôt l'application d'une stratégie de segmentation par étapes basée sur un algorithme de fusion de classes. Cela consiste en l'utilisation d'un processus itératif de fusion et d'affectation des vecteurs de mouvement. En ce sens, l'estimation du flot optique est d'abord divisée en blocs rectangulaires, auxquels sont respectivement affectés les modèles de mouvement dont les propriétés affines minimisent l'erreur résiduelle. Par la suite, à chaque itération, les blocs connexes se ressemblant, c'est-à-dire ceux dont la mesure de distance entre leur modèle de mouvement respectif est en deçà d'un certain seuil T_s, sont fusionnés afin de former des classes. Ces dernières héritent respectivement des propriétés affines du modèle de mouvement de l'un des blocs qui les composent selon un critère de minimisation de l'erreur résiduelle. Simultanément, les vecteurs de mouvement qui satisfont un critère de similitude sont affectés à la classe appropriée. Ce processus se termine lorsque tous les vecteurs ont été affectés ou bien lorsqu'un seuil T_s maximal préalablement fixé est atteint. Une telle stratégie permet de déterminer de manière automatique le nombre de classes devant être créées. Elle permet également de contrer certains problèmes de segmentation excessive ou insuffisante.

En supposant que la caméra se déplace perpendiculairement à l'axe optique, nous montrerons qu'une estimation précise du mouvement 2D peut être directement segmentée, ce qui permet d'éviter d'estimer les propriétés affines du mouvement. Un tel algorithme présente donc deux avantages majeurs. Premièrement, il est simple et rapide. Deuxièmement, les différentes couches sont caractérisées par les vecteurs qu'elles contiennent et non par les propriétés affines du modèle de mouvement. Comme nous l'avons mentionné à la section 2.1, plus un objet est loin, plus son déplacement apparent sera faible, et inversement. Par conséquent, la longueur des vecteurs au sein d'une couche donnée pourra être directement utilisée pour déterminer la profondeur relative de cette couche par rapport aux autres. Préalablement à la mise en oeuvre d'un tel algorithme, il importe toutefois de bien comprendre la notion de flot optique et de connaître les forces et faiblesses des algorithmes associés.

2.3 Définition du flot optique

Le flot optique fut originellement défini comme étant le mouvement apparent de l'intensité lumineuse de l'image [20]. Dans la plupart des cas, un changement d'intensité peut être associé à un mouvement 3D d'un objet de la scène et/ou de la caméra. Cependant, cette relation entre le mouvement et les changements d'intensité est parfois fausse. Par exemple, une sphère parfaitement lisse qui pivote autour d'un axe passant par son centre ne provoque aucun changement d'intensité lumineuse même s'il existe effectivement un mouvement. De plus, une modification de l'éclairage d'une scène statique constitue un exemple dans lequel on retrouve un changement dans l'intensité, mais aucun mouvement. Afin de généraliser l'interprétation du flot optique, une définition constituant une représentation complète des variations géométriques et radiométriques des séquences d'images a récemment été proposée [32]. Selon cette définition, le flot optique décrit la transfor-

mation apparente, et non le mouvement apparent, de l'intensité lumineuse. Cela inclut simultanément les variations géométriques (e.g. mouvement apparent) et radiométriques (e.g. modifications des conditions d'illumination).

Le flot optique peut donc permettre d'estimer le mouvement 2D (direction et vitesse) de la majorité des pixels de l'image. Ce dernier correspond à la projection sur le plan image du mouvement 3D relatif des objets d'une scène par rapport au capteur (caméra). Il est possible d'estimer ce mouvement sous forme de vitesse instantanée ou de déplacement discret. Une telle estimation peut être utilisée dans différents domaines d'application du traitement et de l'analyse d'images. À titre d'exemples, elle peut servir à l'estimation du mouvement 3D de la caméra [2], à la reconstruction 3D de scènes réelles [19, 56], à la compression de séquences vidéo [29, 58], à la mesure de disparité stéréo [42] ou encore à la segmentation d'images [60, 58].

2.3.1 Principes de base

D'une part, selon la définition originale du flot optique, en considérant les variations radiométriques en tant qu'exceptions, l'estimation du mouvement 2D est équivalente à l'estimation du mouvement apparent de l'intensité lumineuse. Soit un pixel P, de coordonnées (x,y), ayant une intensité $E(x,y,t)$ au temps t, cela équivaut donc à déterminer quel pixel P' au temps $t+\delta t$ a une intensité semblable à celle de P. En se basant sur la translation, nous pouvons poser $P' = (x+\delta x, y+\delta y)$ et déduire la relation à l'origine des algorithmes d'estimation du flot optique :

$$E(x+\delta x, y+\delta y, t+\delta t) = E(x,y,t). \tag{2.1}$$

Cette équation met en évidence l'objectif visé, c'est-à-dire le calcul des déplacements δx et δy et/ou des vitesses respectives $u = \frac{\delta x}{\delta t}$ et $v = \frac{\delta y}{\delta t}$.

D'autre part, selon la définition récemment proposée par Negahdaripour [32], la généralisation du modèle du flot optique est donnée par:

$$E(x+\delta x,y+\delta y,t+\delta t) = E(x,y,t) + \delta E(x,y,t), \tag{2.2}$$

où la transformation radiométrique $\delta E(x,y,t)$ est explicitement définie en fonction d'un facteur multiplicatif $\delta m(x,y,t)$ et d'un décalage $\delta c(x,y,t)$:

$$\delta E(x,y,t) = \delta m(x,y,t)E(x,y,t) + \delta c(x,y,t).$$

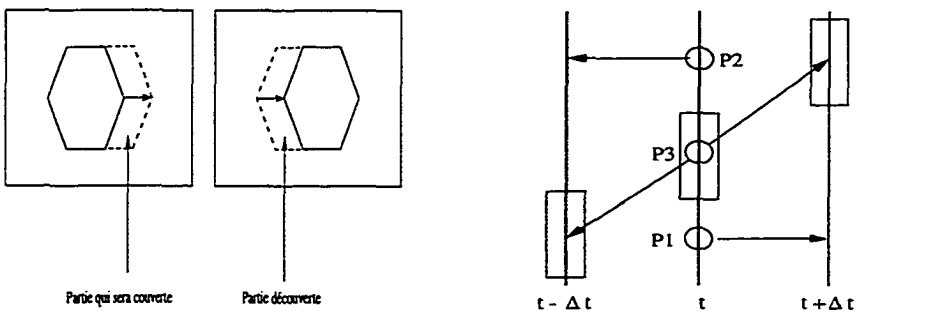
Cette définition suggère qu'un modèle général permet de calculer les composantes géométriques (δx,δy) et radiométriques (δm et δc) du flot optique. Notons également qu'elle englobe la définition originale du flot optique ($\delta E = 0$). Étant donné que celle-ci a été proposée après que nous ayions effectué ce travail, nous ne la considérerons pas dans la suite de ce chapitre. Nous supposerons donc que les variations radiométriques sont faibles et qu'elles n'influencent et n'inhibent pas le mouvement apparent de l'intensité lumineuse associé au mouvement 3D. Cette hypothèse est d'ailleurs à la base des algorithmes traditionnels d'estimation du flot optique (Section 2.4).

2.3.2 Problèmes afférents

Le calcul du mouvement apparent, tel que présenté, manque de fiabilité dans certaines situations. De plus, nous pouvons le qualifier de problème mal posé, puisqu'il ne garantit pas l'obtention d'une solution unique. En fait, il implique trois problèmes majeurs. Premièrement, plusieurs points du voisinage d'un point P ont habituellement une intensité semblable à celle de P. Il peut ainsi y avoir plus d'un pixel pouvant satisfaire la relation (2.1). Par exemple, considérons une séquence d'images contenant une surface en mouvement dont l'intensité est parfaitement uniforme. Un point quelconque de cette surface peut alors être associé à n'importe quel autre point de la surface lors de l'application

de l'équation (2.1). Par conséquent, le flot optique calculé peut prendre plusieurs valeurs et fournir des résultats incohérents. Pour éviter une telle situation, nous supposons généralement que la valeur initiale du mouvement est nulle. Ainsi, lors de l'estimation du mouvement, les pixels qui subissent réellement un déplacement et qui n'appartiennent pas à une région uniforme se verront attribuer une valeur de flot optique différente de zéro. Il semble donc évident que le flot optique ne puisse pas toujours être déterminé de manière unique à partir de l'information locale.

Deuxièmement, la présence d'occlusions occasionne des erreurs au niveau de la mise en correspondance des points P et P'. Comme le montre la figure 2.2a, le terme occlusion représente les parties de surface couvertes/découvertes suite au mouvement d'un objet qui occupe une partie du champ de vision [55]. Supposons un point P_1. S'il apparaît lors



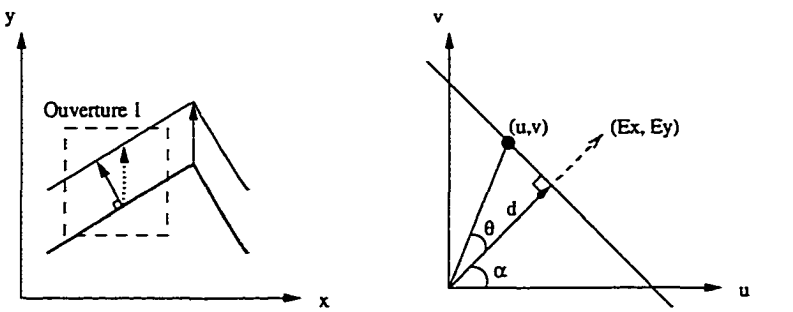
a. Exemples d'occlusions b. Mise en correspondance

Figure 2.2 – *Problème d'occlusions*

du passage de l'image $t-\Delta t$ à l'image t (figure 2.2b), il est alors inadéquat de tenter une mise en correspondance de ce point avec un point de l'image $t-\Delta t$. De même, si un point P_2 disparaît lors du passage de l'image t à l'image $t+\Delta t$, la mise en correspondance de ce point avec un point de l'image $t+\Delta t$ n'a aucun sens. Il est donc évident que la mise en correspondance des points sans considération des occlusions entraîne des erreurs dans le calcul du flot optique. Or, puisque les occlusions proviennent de la présence de mouvement, leur détection peut être considérée comme étant directement reliée à l'estimation

du mouvement. Par conséquent, nous nous retrouvons en présence du problème de l'oeuf et de la poule.

Finalement, tel qu'illustré par la figure 2.3a, il existe un problème d'ouverture. Dans cet



a. Exemples d'ouverture b. Droite de contrainte

Figure 2.3 – *Problème d'ouverture*

exemple, le coin d'un objet se déplace dans la direction de l'axe des y. Si nous estimons le mouvement à partir d'une fenêtre locale (*Ouverture 1* de l'image 2.3a), il est alors impossible de déterminer si l'objet se déplace vers le haut ou perpendiculairement à son contour. En d'autres termes, l'analyse de la relation (2.1) montre clairement que nous sommes en présence d'un système d'équations comportant une équation à deux inconnues (u et v ou encore δx et δy). Cela signifie donc qu'il existe une infinité de solutions et que toutes ces solutions appartiennent à une même droite, comme le montre la figure 2.3b. Étant donné que l'angle θ entre le vecteur vitesse (u,v) et le vecteur gradient (E_x,E_y) est inconnu, il est impossible de déterminer la composante le long du vecteur perpendiculaire à (E_x,E_y). Le problème d'ouverture met donc en évidence le fait que l'estimation du flot optique à partir de l'information locale ne mène pas à une solution unique. Ce manque d'unicité dans l'estimation du mouvement implique donc l'ajout de contraintes supplémentaires permettant de restreindre les valeurs possibles de (u,v). Afin de solutionner ce problème, plusieurs approches ont été élaborées.

2.4 Principaux algorithmes du flot optique

Il existe plusieurs algorithmes d'estimation du mouvement apparent. La plupart se basent sur la relation (2.1). Toutefois, leur spécificité réside au niveau des contraintes utilisées pour restreindre le problème relié à l'absence d'une solution unique. Ces algorithmes appartiennent généralement à l'une des trois principales catégories suivantes :

1. Algorithmes basés sur la différentiation
2. Algorithmes basés sur la mise en correspondance
3. Algorithmes basés sur le spectre

Il est à noter qu'il existe une quatrième catégorie. Celle-ci contient les algorithmes basés sur l'utilisation des images de contours. Ces algorithmes utilisent également la différentiation, la mise en correspondance ou le spectre. Leur principale différence se situe au niveau de l'utilisation d'images de contours au lieu d'images d'intensité. Par conséquent, nous ne tiendrons pas compte explicitement de cette catégorie dans ce qui suit.

2.4.1 Algorithmes basés sur la différentiation

Le calcul du vecteur vitesse (u,v) à partir des dérivées spatio-temporelles de l'image d'intensité constitue le principe à la base de cette catégorie d'algorithmes. Les premières instances de cette technique utilisaient les dérivées du premier ordre et se basaient sur la relation (2.1). En supposant que l'intensité ne varie pas dans le temps selon la direction du mouvement, la dérivée de l'intensité par rapport au temps dans cette direction respecte la relation :

$$\frac{\delta E}{\delta x}\frac{dx}{dt} + \frac{\delta E}{\delta y}\frac{dy}{dt} + \frac{\delta E}{\delta t} = 0,$$

d'où l'équation de contrainte du gradient (ou équation de contrainte du flot optique) proposée par Horn et Schunck [20] :

$$E_x u + E_y v + E_t = 0, \tag{2.3}$$

où $u = \frac{dx}{dt}$ et $v = \frac{dy}{dt}$. Dans l'espace (u,v), cette équation représente la droite située à une distance $d = E_t / \sqrt{E_x^2 + E_y^2}$ de l'origine et orientée selon un angle $\alpha = \arctan(E_y / E_x)$ (voir figure 2.3b).

Grâce à cette équation, il est possible de calculer la valeur de la composante du vecteur (u,v) le long du vecteur gradient. Cependant, cela ne règle pas le problème du manque d'unicité de la solution énoncé à la section 2.3.2. Pour résoudre le système d'équations, les algorithmes présentés utilisent donc des contraintes supplémentaires sur u et v.

Algorithme de Horn et Schunck

Une première technique utilisée pour contraindre u et v consiste à considérer les estimations des composantes de vitesse d'un voisinage local à travers le temps et l'espace. Dans cette optique, l'algorithme de Horn et Schunk [20] utilise une contrainte de lissage global. C'est-à-dire qu'il suppose que le flot optique varie habituellement de manière lisse pour une région donnée. Il suffit donc de pénaliser les sauts brusques dans les fonctions $u(x,y)$ et $v(x,y)$ en minimisant l'intégrale du module au carré du gradient du flot optique :

$$e_s = \int \int ((u_x^2 + u_y^2) + (v_x^2 + v_y^2)) dx dy.$$

Cependant, dans la pratique, la minimisation de e_s conjointement avec l'application directe de la contrainte du flot optique (équation (2.3)) ne suffisent pas. En effet, plusieurs causes, dont l'imprécision numérique du calcul des dérivées, empêchent le respect de la relation (2.3). Pour cette raison, il est préférable de rechercher une solution minimisant aussi l'erreur suivante :

$$e_c = \int \int (E_x u + E_y v + E_t)^2 dx dy.$$

Selon cette approche, le problème du flot optique revient donc à minimiser e_c avec e_s comme contrainte, c'est-à-dire $e_s + \lambda e_c$:

$$\int\int (u_x^2 + u_y^2) + (v_x^2 + v_y^2) + \lambda(E_x u + E_y v + E_t)^2 dxdy, \tag{2.4}$$

où λ est un paramètre qui pondère l'erreur par rapport au lissage. La recherche du minimum relève du calcul variationnel et ne sera pas explicitée dans ce document. Pour plus de détails, se référer à [20] ou [19].

Au niveau numérique, la résolution de ce système d'équations peut être effectuée grâce aux équations itératives :

$$u^k = u_{av}^{k-1} - E_x \frac{E_x u_{av}^{k-1} + E_y v_{av}^{k-1} + E_t}{\lambda^2 + E_x^2 + E_y^2} \tag{2.5}$$

et

$$v^k = v_{av}^{k-1} - E_y \frac{E_x u_{av}^{k-1} + E_y v_{av}^{k-1} + E_t}{\lambda^2 + E_x^2 + E_y^2} \tag{2.6}$$

où u_{av}^k et v_{av}^k représentent respectivement les valeurs moyennes de u^k et v^k dans un voisinage donné et $k = 1 \cdots K$ le nombre d'itérations. Il est à noter que les valeurs initiales u^0 et v^0 sont toutes deux égales à 0 (se référer à la section 2.3.2).

Algorithme de Lucas et Kanade

Une seconde technique utilisée pour contraindre u et v se base sur l'hypothèse que le vecteur $\mathbf{v} = (u,v)^T$ est constant dans un voisinage spatial donné (Ω). L'algorithme de Lucas et Kanade applique cette contrainte en minimisant la somme pondérée des moindres carrés suivante :

$$\sum_{(x,y)\in\Omega} W^2(x,y)(E_x u + E_y v + E_t)^2, \tag{2.7}$$

où $W(x,y)$ représente une fonction fenêtre qui définit le voisinage. La solution de l'équation (2.7) est donnée par [3] :

$$A^T W^2 A\mathbf{v} = A^T W^2 \mathbf{b}, \tag{2.8}$$

où pour n points $p_i = (x_i,y_i) \in \Omega$ au temps t :

$$A = [\nabla E(p_1), \ldots, \nabla E(p_n)]^T$$
$$W = diag[W(p_1), \ldots, W(p_n)]$$
$$\mathbf{b} = -[E_t(p_i), \ldots, E_t(p_n)]^T,$$

avec $\nabla E = (E_x,E_y)$. Ainsi, l'estimation du mouvement nous est directement fournie par $\mathbf{v} = [A^T W^2 A]^{-1} A^T W^2 \mathbf{b}$ lorsque $A^T W^2 A$ est inversible.

Algorithme de Nagel

Certaines techniques emploient les dérivées du second ordre. Plus précisément, elles utilisent le hessien de l'image pour contraindre u et v :

$$\begin{bmatrix} E_{xx} & E_{yx} \\ E_{xy} & E_{yy} \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} + \begin{bmatrix} E_{tx} \\ E_{ty} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \tag{2.9}$$

Cette équation peut être obtenue à partir de (2.1) ou bien en supposant que le gradient d'un point donné (∇E) ne change pas dans le temps selon la trajectoire du mouvement ($d\nabla E/dt = 0$). Cependant, une telle supposition implique l'absence des déformations d'intensité telles que la rotation ou la dilatation. Notons qu'une telle restriction est évidemment plus stricte que celle présentée par l'équation (2.3). En se basant sur l'hypothèse que $d\nabla E/dt = 0$, le calcul de la vitesse peut être effectué en utilisant conjointement les équations (2.3) et (2.9). Cependant, les dérivées numériques sont sensibles au bruit. Ainsi, les composantes de vitesse estimées par les techniques utilisant les dérivées du second ordre sont habituellement moins précises que celles n'utilisant que les dérivées du premier ordre.

L'une des premières techniques basées sur les dérivées du second ordre fut proposée par Nagel [31]. Comme alternative à l'utilisation de la contrainte proposée par Horn et Schunck (2.4), Nagel suggère plutôt l'application d'un lissage global orienté dans la direction perpendiculaire au gradient. Le but principal de cette approche consiste en une meilleure gestion des occlusions. Selon Nagel, il est possible d'atténuer les variations du flot optique dans la direction perpendiculaire au gradient en minimisant :

$$\int\int (E_x u + E_y v + E_t)^2 + \frac{\alpha^2}{E_x^2 + E_y^2 + 2\beta} \times [(u_x E_y - u_y E_x)^2 + (v_x E_y - v_y E_x)^2 + \delta(u_x^2 + u_y^2 + v_x^2 + v_y^2)] dx dy, \quad (2.10)$$

où $\mathbf{v} = (u,v)$, $\nabla E = (E_x,E_y)$, β et α sont des paramètres de l'algorithme et $\delta(x)$ représente la distribution de Dirac.

La résolution de ce système d'équations peut être effectuée grâce aux équations itératives :

$$u^k + 1 = \xi(u^k) - \frac{E_x(E_x\xi(u^k) + E_y\xi(v^k) + E_t)}{E_x^2 + E_y^2 + \alpha^2} \quad (2.11)$$

et

$$v^k + 1 = \xi(v^k) - \frac{E_y(E_x\xi(u^k) + E_y\xi(v^k) + E_t)}{E_x^2 + E_y^2 + \alpha^2}. \quad (2.12)$$

Dans ces équations, $k = 1 \cdots K$ représente le nombre d'itérations et $\xi(u^k)$ et $\xi(v^k)$ sont données par :

$$\xi(u^k) = u_{av}^k - 2E_x E_y u_{xy} - \mathbf{q}^T(\nabla u^k)$$
$$\xi(v^k) = v_{av}^k - 2E_x E_y v_{xy} - \mathbf{q}^T(\nabla v^k)$$

où

$$\mathbf{q} = \frac{1}{E_x^2 + E_y^2 + 2\beta} \nabla E^T \left(\begin{bmatrix} E_{yy} & -E_{xy} \\ -E_{xy} & E_{xx} \end{bmatrix} + 2 \begin{bmatrix} E_{xx} & E_{xy} \\ E_{xy} & E_{yy} \end{bmatrix} W \right).$$

u_{av}^k et v_{av}^k représentent respectivement les valeurs moyennes de u^k et v^k dans un voisinage donné et W est une matrice de pondération :

$$W = \frac{1}{(E_x^2 + E_y^2 + 2\beta)} \begin{bmatrix} E_y^2 + \beta & -E_x E_y \\ -E_x E_y & E_x^2 + \beta \end{bmatrix}.$$

L'algorithme proposé consiste donc en une implantation des équations (2.11) et (2.12). L'analyse plus approfondie de la démarche utilisée par Nagel ne fait pas partie du cadre de ce document. Il est toutefois possible de trouver de plus amples informations en se référant à [31].

Algorithme de Uras, Girosi, Verri et Torre

Le quatrième algorithme basé sur la différentiation est celui proposé par Uras *et al.* [57]. Tout comme pour l'algorithme de Nagel, la dérivée du second ordre est utilisée pour contraindre u et v. Sachant qu'il existe une solution $\mathbf{v} = (u,v)$ dans tous les cas où le hessien (H) de $E(x,y,t)$ est inversible, cette approche se base sur la recherche d'une solution locale de l'équation (2.9). En pratique, pour assurer plus de robustesse, l'image est divisée en régions de 8×8 pixels. Pour chaque région, les 8 estimations qui satisfont le mieux la contrainte :

$$||(\nabla \mathbf{v})^T \nabla E|| \ll ||\nabla E_t|| \tag{2.13}$$

sont retenues. Parmi celles-ci, l'estimation ayant le plus petit nombre de conditionnement $(\kappa(H))$ détermine la vitesse pour la région 8×8. Le nombre de conditionnement d'une matrice A est donné par $\kappa(A) = \frac{\lambda_{max}}{\lambda_{min}}$ où les λ correspondent aux valeurs propres de la matrice. À partir de ce nombre, il est possible de déterminer si la matrice donnée est bien ou mal conditionnée. Mentionnons que, selon Barron *et al.* [3], l'utilisation de $det(H)$ au lieu de $\kappa(H)$, comme mesure de confiance des estimations, augmente la fiabilité. Cette affirmation semble toutefois provenir de résultats expérimentaux et aucune démonstration n'est présentée.

2.4.2 Algorithmes basés sur la mise en correspondance

Les approches basées sur la mise en correspondance (corrélation) définissent le mouvement en tant que déplacement $\mathbf{d} = (\delta x,\delta y)$. Le principe consiste à rechercher le déplacement $\mathbf{d}$ fournissant la meilleure mise en correspondance entre des fenêtres de l'image à différents moments. Une telle recherche équivaut donc à maximiser une mesure de similarité, telle que la corrélation, ou à minimiser une mesure de distance, telle que la somme des carrés de différences (SSD) [3] :

$$SSD_{t,t+1}(x,y;\mathbf{d}) = \sum_{j=-n}^{n}\sum_{i=-n}^{n} W(i,j) \times [E(x+i,y+j,t) - E(x+\delta x+i,y+\delta y+j,t+1)]^2, \tag{2.14}$$

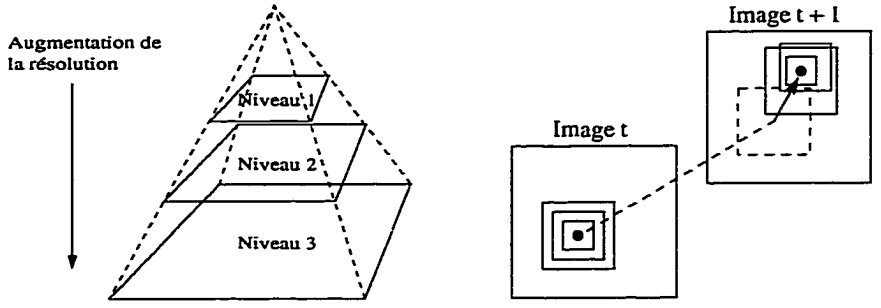
où W est une fenêtre 2D définissant le voisinage de (x,y) et $\delta x,\delta y$ sont des valeurs entières.

En analysant de plus près l'équation (2.14), nous remarquons qu'il existe une similitude entre cette technique et celle de Lucas et Kanade. En effet, la soustraction présente dans cette équation peut être vue comme une approximation de dérivées temporelles de $E(x,y,t)$.

Afin de mieux illustrer le principe d'utilisation de la mise en correspondance pour l'estimation du mouvement, la sous-section suivante présente l'algorithme d'Anandan.

Algorithme d'Anandan

L'approche proposée par Anandan [1] est basée sur l'utilisation de pyramides Laplaciennes et sur une stratégie de mise en correspondance multi-résolutions. Selon l'auteur, l'utilisation de la représentation hiérarchique (multi-résolutions) des images, sous forme de pyramides Laplaciennes (figure 2.4a), fournit de meilleurs résultats. Le principe de base consiste à utiliser l'équation (2.14) pour estimer successivement le mouvement à chaque niveau hiérarchique, en commençant avec le niveau dont la résolution est la plus



a. Pyramide Laplacienne b. Mise en correspondance

Figure 2.4 – *Principe proposé par Anandan*

basse. Les estimations des niveaux de faible résolution servent d'estimations initiales pour les niveaux de haute résolution. Comme le montre la figure 2.4b, cela est équivalent à l'utilisation de fenêtres W de plus en plus petites lors de l'étape de mise en correspondance. C'est-à-dire que plus le niveau traité possède une haute résolution, plus la fenêtre utilisée est relativement petite, puisque l'estimation initiale est de plus en plus précise. Notons que l'utilisation de l'équation (2.14) ne fournit pas une solution $\mathbf{v} = (u,v)$ unique. Anandan propose donc l'emploi d'une contrainte de lissage qui se traduit par la minimisation de :

$$\int\int (u_x^2 + u_y^2 + v_x^2 + v_y^2) + c_{max}(\mathbf{v} \cdot \mathbf{e}_{max} - \mathbf{v}_0 \cdot \mathbf{e}_{max})^2 + c_{min}(\mathbf{v} \cdot \mathbf{e}_{min} - \mathbf{v}_0 \cdot \mathbf{e}_{min})^2 dxdy, \tag{2.15}$$

où $\cdot$ est un produit scalaire et

$$\begin{aligned} c_{max} &= \frac{C_{max}}{k_1 + k_2 S_{min} + k_3 Cmax} \\ c_{min} &= \frac{C_{min}}{k_1 + k_2 S_{min} + k_3 Cmin} \end{aligned}$$

c_{min} et c_{max} représentent des mesures de confiance obtenues à partir des courbures principales (C_{min} et C_{max}) du minimum S_{min} de la surface SSD. $\mathbf{e}_{min}$ et $\mathbf{e}_{max}$ représentent les directions respectives de C_{min} et C_{max}. v_0 est la valeur initiale du déplacement, tandis que k_1, k_2, k_3 sont des constantes de normalisation.

Au niveau numérique, la résolution de ce système d'équations peut être effectuée grâce à l'équation itérative :

$$\mathbf{v}^{k+1} = \mathbf{v}_{av}^{k} + \frac{c_{max}}{c_{max}+1}[(\mathbf{v}_0 - \mathbf{v}_{av}^{k}) \cdot \mathbf{e}_{max}]\mathbf{e}_{max} + \frac{c_{min}}{c_{min}+1}[(\mathbf{v}_0 - \mathbf{v}_{av}^{k}) \cdot \mathbf{e}_{min}]\mathbf{e}_{min} \quad (2.16)$$

où $k = 1 \cdots K$ est le nombre d'itérations et $\mathbf{v}_{av}^{k}$ représente la valeur moyenne des $\mathbf{v}^{k}$ calculée dans un voisinage avec le masque :

$$\frac{1}{4}\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}.$$

Il est à noter que l'utilisation d'images multi-résolutions n'est pas limitée à cette approche. Elle peut également être combinée à certains autres algorithmes appartenant à l'une des quatre catégories présentées.

2.4.3 Algorithmes basés sur le spectre

La dernière catégorie d'algorithmes se base sur le spectre (amplitude ou phase) des images dans le domaine de Fourier. Au niveau de l'utilisation du spectre d'amplitude, la transformée de Fourier d'une image subissant une translation 2D peut s'écrire comme :

$$\widehat{E}(\mathbf{k},\omega) = \widehat{E}_0(\mathbf{k})\delta(\omega + \mathbf{v}^T\mathbf{k}), \quad (2.17)$$

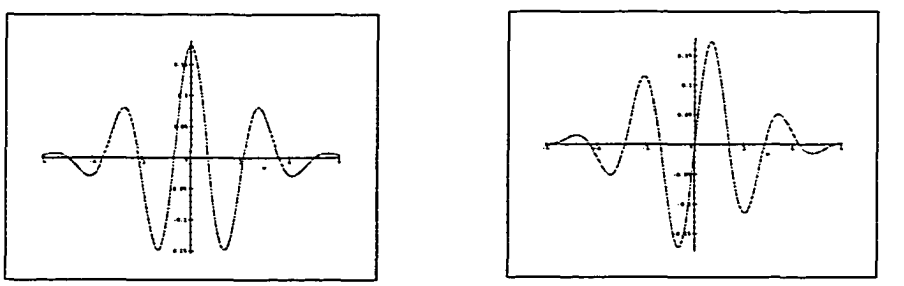
où $\widehat{E}_0(\mathbf{k})$ est la transformée de Fourier de $E(\mathbf{x},0)$, $\delta(k)$ la distribution de Dirac, ω la fréquence temporelle et $\mathbf{k} = (k_x,k_y)$ la fréquence spatiale. D'après cette relation, lorsque $\widehat{E}(\mathbf{k},\omega)$ est non nulle, elle se retrouve dans le plan $\omega + uk_x + vk_y = 0$. Cela correspond à l'équation d'un plan passant par l'origine dans le domaine des fréquences [3]. En d'autres termes, les solutions possibles de ce système appartiennent à une même droite, comme celle présentée donc la figure 2.3.

Selon Barron *et al.* [3], il a été démontré que la plupart des algorithmes basés sur le spectre d'amplitude sont équivalents à ceux basés sur la mise en correspondance ou à celui proposé par Lucas et Kanade. Pour cette raison, nous n'approfondirons pas davantage cette approche. La référence [3] contient de plus amples informations.

Au niveau de l'utilisation du spectre de phase, certains algorithmes définissent la vitesse à partir de la phase obtenue suite à l'utilisation de filtres passe-bandes. À titre d'exemple, voici l'algorithme de Fleet et Jepson.

Algorithme de Fleet et Jepson

Fleet et Jepson [15] définissent les composantes de vitesse en fonction du mouvement instantané normal aux courbes de niveau de la phase. Les filtres passe-bandes sont utilisés pour décomposer le signal selon l'échelle, la vitesse et l'orientation. Plus précisément, cet algorithme emploie les filtres de Gabor [35] (figure 2.5). Chaque image résultante de la



a. Partie réelle $\left(\frac{1}{\sqrt{2\pi}\sigma}\exp[\frac{-x^2}{2\sigma^2}]\cos(\omega x)\right)$

b. Partie imaginaire $\left(\frac{1}{\sqrt{2\pi}\sigma}\exp[\frac{-x^2}{2\sigma^2}]\sin(\omega x)\right)$

Figure 2.5 – *Exemple de filtre de Gabor*

convolution avec un tel filtre est complexe et peut être écrite sous la forme :

$$R(x,y,t) = \rho(x,y,t)\exp[i\phi(x,y,t)], \tag{2.18}$$

où $\rho(x,y,t)$ et $\phi(x,y,t)$ représentent respectivement l'amplitude et la phase de $R(x,y,t)$.

Les composantes de vitesse dans la direction normale aux courbes de niveau de la phase sont données par $v_n = s\mathbf{n}$, où la vitesse normale et la direction proviennent de :

$$s = \frac{-\phi_t(x,y,t)}{\sqrt{(\phi_x(x,y,t)^2 + \phi_y(x,y,t)^2)}} \tag{2.19}$$

et

$$\mathbf{n} = \frac{\nabla\phi(x,y,t)}{\sqrt{(\phi_x(x,y,t)^2 + \phi_y(x,y,t)^2)}}, \tag{2.20}$$

où $\nabla\phi(x,y,t) = (\phi_x(x,y,t),\phi_y(x,y,t))^T$. D'après cette équation, il est clair que nous sommes en présence d'une technique différentielle appliquée à l'image de phase. L'utilisation de la phase plutôt que l'amplitude ou l'intensité se justifie par sa robustesse. Cette information est moins sensible aux variations de contraste, d'échelle, d'orientation et de vitesse. Cependant, Fleet et Jepson ont montré que cette information peut également être instable dans les régions voisines des singularités de phase. Il est toutefois possible de les détecter grâce à l'utilisation de contraintes sur la fréquence instantanée et les variations d'amplitude dans l'espace temps. Une contrainte supplémentaire sur l'amplitude est également utilisée par assurer un rapport signal/bruit raisonnable.

Pour le calcul des dérivées partielles de la phase, Fleet et Jepson suggèrent l'utilisation de la formule suivante :

$$\nabla\phi(x,y,t) = \frac{Im[R^*(x,y,t) \cdot \nabla R(x,y,t)]}{\rho^2(x,y,t)}, \tag{2.21}$$

où $\cdot$ est un produit scalaire et R^* le conjugué complexe de R. Cette relation permet de contourner les problèmes reliés aux discontinuités de phase.

À partir des composantes de vitesse estimées grâce aux différents filtres, un modèle linéaire de vitesse est associé à chaque région locale. Les estimations qui satisfont les contraintes de stabilité et du rapport signal/bruit sont groupées par voisinage de 5×5. Pour chaque groupe ainsi créé, le meilleur modèle de vitesse linéaire est déterminé en

appliquant une minimisation des moindres carrés. Une analyse plus approfondie de la démarche utilisée par Fleet et Jepson est disponible dans [15].

2.5 Évaluation comparative des algorithmes

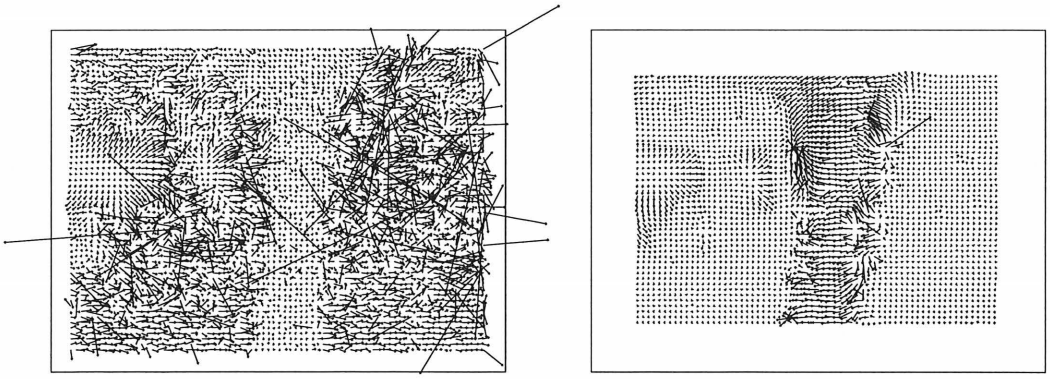
Dans un contexte de segmentation d'images en couches, les résultats fournis par la méthode d'estimation du flot optique sont primordiaux. Ils ont une influence directe sur la qualité de la segmentation résultante. Il est donc nécessaire d'évaluer les algorithmes existants et de choisir celui conduisant à une segmentation adéquate. Barron *et al.* [3] ont effectuée une évaluation quantitative de la densité et de l'erreur d'orientation moyenne du flot optique estimé par différents algorithmes. Notons que celle-ci a été réalisée sans aucune référence à une application donnée. Nous proposons donc une évaluation subjective des 6 algorithmes présentés à la section 2.4 dans un contexte de segmentation d'images en couches. Plus précisément, nous évaluerons les résultats de ces algorithmes selon trois critères. Le premier concerne le degré de lissage de la solution au sein d'une région de mouvement homogène. Le second est la netteté des frontières entre des régions hétérogènes. Le dernier critère est l'amplitude relative des déplacements. Afin de pouvoir comparer les résultats obtenus, les choix des paramètres de chaque algorithme sont effectués de manière à obtenir les meilleurs résultats. Mentionnons que pour effectuer ces tests comparatifs, nous avons récupéré le code source utilisé par Barron *et al.* La séquence d'images utilisée (figure 2.6) illustre le mouvement relatif des objets par rapport à la caméra lorsque cette dernière subit une translation vers la droite.

Les estimations résultantes du flot optique sont représentées par des diagrammes d'aiguilles. L'orientation et le module des vecteurs correspondent respectivement à la direction et à l'amplitude des déplacements. D'abord, les figures 2.7a et 2.7b illustrent le flot optique calculé par l'algorithme de Horn et Schunck (équations (2.5) et (2.6)). Les

a. Image au temps $t = 0$ b. Image au temps $t = 9$ c. Image au temps $t = 20$

Figure 2.6 – *Exemple d'images de la séquence vidéo employée*

paramètres de l'algorithme sont la pondération $\lambda = 2$ et le nombre maximal d'itérations $K = 100$. La version originale de cet algorithme a fourni une estimation non lisse du flot optique. Nous remarquons également la présence de nombreuses valeurs aberrantes et une amplitude trop faible au niveau de l'arbre. En revanche, la version modifiée, c'est-à-dire celle dont le prétraitement consistait en un lissage spatio-temporel gaussien des images, a permis l'obtention d'une solution plus lisse. L'échelle utilisée est $\sigma = 1.5$. L'homogénéité des régions est donc accrue et certaines frontières apparaissent. Cette solution manque toutefois légèrement de précision, principalement au niveau de l'amplitude du mouvement des fleurs se trouvant entre l'arbre et la maison qui est trop faible. Cela occasionne un mauvais découpage des frontières de ces régions.



a. Version originale b. Avec prétraitement

Figure 2.7 – *Estimation du mouvement avec l'algorithme de Horn et Schunck*

Pour sa part, l'algorithme de Lucas et Kanade semble fournir des résultats plus ou moins fiables (figure 2.8). Ceux-ci ont été obtenus avec une fenêtre 5×5 (équation (2.7)). Nous distinguons la présence des frontières des régions de mouvement associées respectivement à l'arbre, aux fleurs, à la maison et au ciel. Cette segmentation implicite est principalement reliée au principe de constance dans chaque voisinage qui est à la base de cet algorithme. Cependant, nous constatons un manque d'homogénéité au niveau de certaines régions telles que l'arbre et la maison en arrière-plan. De plus, les amplitudes

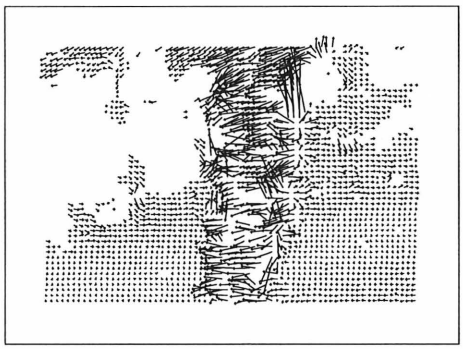


Figure 2.8 – *Estimation du mouvement avec l'algorithme de Lucas et Kanade*

relatives des fleurs et de la maison sont incorrectes et ne pourraient donc pas permettre d'ordonner correctement ces deux couches.

Les résultats de l'utilisation de l'algorithme de Nagel sont présentés à la figure 2.9. Les paramètres utilisés sont $\beta = 1$, $\alpha = 0.5$ et un nombre maximal d'itérations $K = 100$ (équations (2.11) et (2.12)). Ces résultats ne sont pas conformes aux attentes. En effet, l'estimation du flot optique proposée par Nagel suppose une meilleure gestion des occlusions et par conséquent des frontières. Cependant, l'estimation résultante est inadéquate aux occlusions. De plus, les régions de mouvement associées aux fleurs, à la maison et au ciel semblent avoir été confondues. Elles ont des amplitudes similaires. Étant donné que cet algorithme utilise des dérivées du premier et du second ordre, l'une des hypothèses pouvant justifier de tels résultats est l'imprécision des dérivées numériques d'ordre

supérieur.

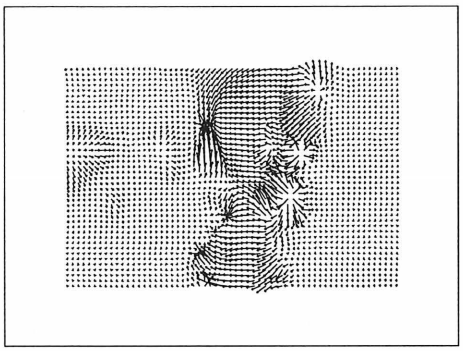


Figure 2.9 – *Estimation du mouvement avec l'algorithme de Nagel*

Le diagramme d'aiguilles du flot optique calculé par l'algorithme de Uras *et al.* (figure 2.10) illustre clairement un problème d'imprécision (équation (2.13)). Cette imprécision se remarque à deux niveaux. D'une part, les mouvements respectifs des fleurs, de la maison et du ciel semblent être confondus. Ils ont quasiment la même amplitude. D'autre part, le mouvement estimé est formé de petites régions carrées qui rendent le découpage des frontières imprécis. Ce regroupement en régions provient du fait que l'algorithme détermine un flot constant pour des régions 8×8 (section 2.4.1).

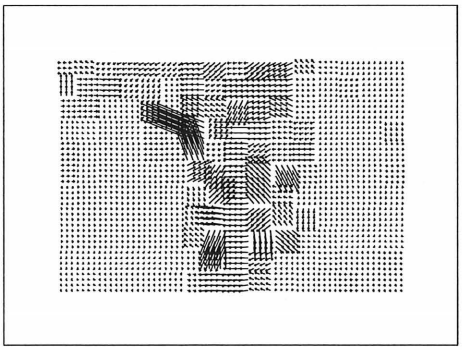


Figure 2.10 – *Estimation du mouvement avec l'algorithme de Uras et al.*

Le cinquième algorithme testé est celui proposé par Anandan. Les paramètres utilisés

sont une fenêtre 3×3 et un nombre maximal d'itérations $K = 10$ (équation (2.16)). Bien que l'amplitude du flot optique résultant (figure 2.11) semble inversement proportionnelle à l'amplitude du flot optique réel, les résultats semblent adéquats dans un contexte de segmentation d'images. En effet, en regardant attentivement le diagramme d'aiguilles, nous distinguons les différences relatives de mouvement (régions homogènes et frontières) entre l'arbre, les fleurs, la maison et le ciel.

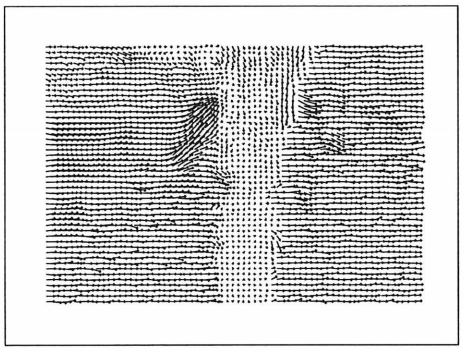


Figure 2.11 – *Estimation du mouvement avec l'algorithme d'Anandan*

Le dernier algorithme testé est celui proposé par Fleet et Jepson. Le diagramme d'aiguilles du flot optique calculé par cet algorithme (figure 2.12) montre un manque de précision au niveau de l'amplitude et de la régularité du mouvement au sein de certaines régions. Cela occasionne une diminution de l'homogénéité et un découpage inadéquat des frontières. De plus, le flot optique n'est pas défini pour tous les points. L'une des causes de ce problème est reliée aux contraintes de cet algorithme relativement aux singularités de phase et au rapport signal/bruit (cf. section 2.4.3).

De manière générale, il semble que l'algorithme de Horn et Schunck avec prétraitement et celui d'Anandan fournissent les meilleurs résultats dans un contexte de segmentation d'images en couches. Mentionnons toutefois que tous ces algorithmes produisent des estimations erronées en présence d'occlusions.

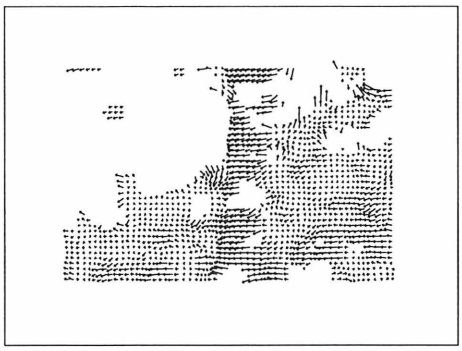


Figure 2.12 – *Estimation du mouvement avec l'algorithme de Fleet et Jepson*

2.6 Algorithme de segmentation en couches

Notre méthode de détermination de l'ordre de profondeur des objets est basée sur une segmentation en couches des images. Pour ce faire, le flot optique est d'abord calculé. Ensuite, les images sont segmentées à partir d'un critère d'homogénéité du flot optique. Voici donc la démarche proposée pour chacune de ces étapes.

Au niveau de l'estimation du flot optique, nous avons choisi d'implanter l'algorithme proposé par Horn et Schunck (section 2.4.1). Cela se traduit par l'application directe des équations itératives (2.5) et (2.6). Après avoir pris connaissance des différentes approches et des résultats des tests comparatifs, nous avons constaté que cet algorithme est le meilleur compromis compte tenu des critères énoncés à la section précédente. La version originale de cet algorithme propose une estimation des dérivées de l'image à partir de quatre différences finies. Par exemple, le calcul de la dérivée partielle en x est donné par :

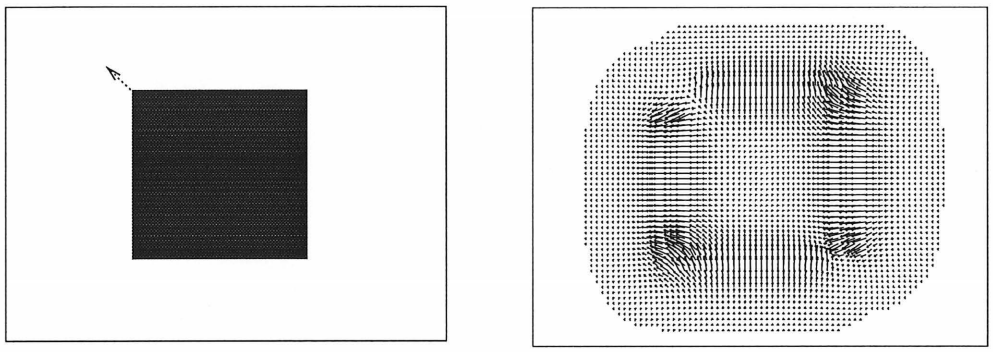
$$\begin{aligned} E_x \approx\ & \tfrac{1}{4}(E(x+1,y,t)+E(x+1,y,t+1)+E(x+1,y+1,t)+E(x+1,y+1,t+1)) \\ & -\tfrac{1}{4}(E(x,y,t)+E(x,y,t+1)+E(x,y+1,t)+E(x,y+1,t+1)). \end{aligned} \tag{2.22}$$

Puisqu'une telle approximation est généralement imprécise, nous croyons que l'emploi d'une approche de différentiation basée sur les propriétés de la convolution et de la distribution gaussienne améliorerait la précision des résultats. Ainsi, la relation (2.22)

peut être remplacée par :

$$E_x = E * g_x \tag{2.23}$$

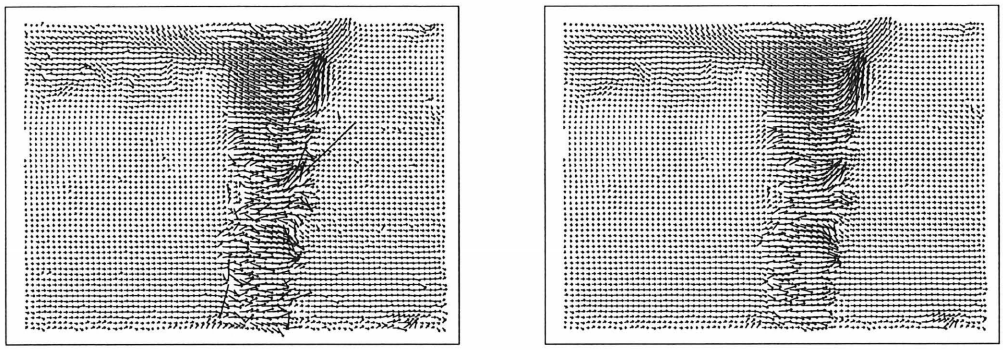
où g représente la gaussienne et $*$ l'opérateur de convolution. Les figures 2.13 et 2.14 montrent des exemples de résultats obtenus avec notre version modifiée de l'algorithme de Horn et Schunck. La première figure illustre le mouvement d'un carré noir se déplaçant vers le coin supérieur gauche de l'image. Les images employées comportent des régions uniformes et non bruitées. Les paramètres utilisés sont $\lambda = 2$ et $K = 32$ (équations (2.5)



a. Image originale b. Mouvement estimé

Figure 2.13 – *Effets des problèmes reliés au flot optique*

et (2.6)). L'estimation calculée illustre clairement les problèmes d'ouverture et de régions uniformes (section 2.3.2). D'une part, les composantes de vitesse perpendiculaires au gradient sont généralement nulles aux points de contours. D'autre part, le flot optique estimé au centre du carré ne correspond pas au mouvement réel. Ces résultats ne remettent toutefois pas en doute la fiabilité de cet algorithme, puisqu'une région uniforme ne contient pas d'information sur le mouvement. Les tests effectués par Barron *et al.* [3] avec la même séquence d'images fournissent d'ailleurs des résultats similaires. La seconde figure illustre les résultats obtenus à partir de la séquence d'images utilisée lors des tests comparatifs (paramètres: $\lambda = 2$ et $K = 32$). La comparaison de ces résultats avec ceux de



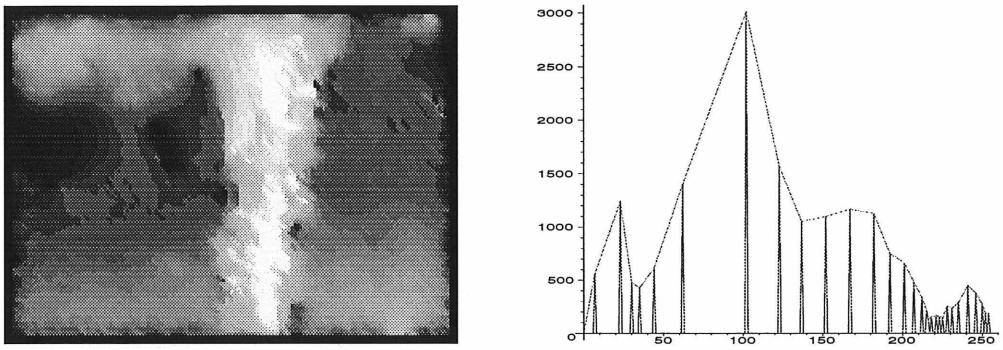
a. Image originale b. Image avec lissage médian

Figure 2.14 – *Résultats de l'algorithme de Horn et Schunck modifié*

la figure 2.7b confirme l'hypothèse selon laquelle l'utilisation d'une technique de différentiation plus précise assure de meilleurs résultats. En effet, en regardant attentivement le diagramme d'aiguilles de l'estimation du flot optique (figure 2.14a), on distingue la présence de régions de mouvement homogène associées respectivement à l'arbre, aux fleurs, à la maison et au ciel. Ce diagramme dénote toutefois un problème majeur. Il contient certaines valeurs aberrantes. L'application d'un lissage médian sur cette estimation permet toutefois d'atténuer ce problème (figure 2.14b). *A priori*, les résultats obtenus semblent donc satisfaisants. En les analysant davantage, nous constatons toutefois la présence d'erreurs principalement au niveau des occlusions. De plus, le lissage effectué lors de la différentiation occasionne certaines erreurs de localisation. Par exemple, on remarque que les régions représentant l'arbre et sa branche supérieure gauche sont trop larges.

Au niveau de la segmentation, un critère d'homogénéité basé uniquement sur la longueur des vecteurs (module) est appliqué à l'estimation du flot optique. Puisque nous supposons que la séquence d'images provient d'une caméra en mouvement dans un environnement statique, l'orientation des vecteurs de déplacement est quasiment constante. Un tel critère s'avère donc suffisant. Plus précisément, l'algorithme de segmentation consiste à calculer l'histogramme du module du flot optique, puis à en extraire les frontières des couches,

c'est-à-dire les minima locaux de l'histogramme lissé. Une technique de segmentation plus robuste pourrait également être utilisée [11], mais cela ne fait pas l'objet de ce travail. La figure 2.15 présente un exemple d'image de modules et l'histogramme associé. Cet exemple a été créé à l'aide de la séquence vidéo présentée à la section 2.5. Cette technique est



a. Image du module b. Histogramme

Figure 2.15 – *Exemple d'image du module du flot optique et de l'histogramme associé*

simple et rapide. Elle nous permet d'obtenir simultanément une segmentation en couches et un ordre de profondeur des couches les unes par rapport aux autres. En effet, plus le module du mouvement est élevé, plus le déplacement est grand et plus la couche se situe près de la caméra, et inversement.

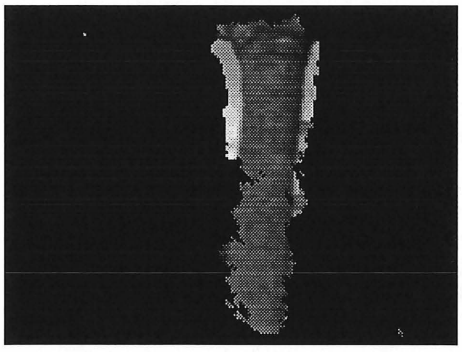
En résumé, soient le paramètre de pondération λ et le nombre maximal d'itérations K, l'algorithme est composé de deux étapes :

1. Estimation du flot optique : application de l'algorithme de Horn et Schunck (équations (2.5) et (2.6)). Les dérivées des images sont calculées en convoluant ces dernières avec les dérivées appropriées de la gaussienne. Les valeurs aberrantes sont éliminées à l'aide d'un filtrage médian.
2. Segmentation des images : extraction ordonnée des minima locaux de l'histogramme du module du flot optique.

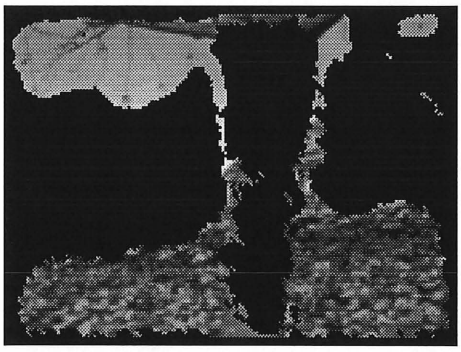
2.7 Évaluation des performances

2.7.1 Résultats expérimentaux

Nous avons implanté et testé l'approche proposée. La séquence d'images utilisée illustre le mouvement relatif des objets par rapport à la caméra lorsque cette dernière subit une translation vers la droite (figure 2.6). Cette séquence vidéo contient au moins quatre couches distinctes de mouvement : l'arbre, les fleurs, la maison et une partie du ciel (là où il n'y a pas d'arbre). La figure 2.16 présente les images au temps $t = 9$ de la segmentation en couches résultante. Tel qu'illustré, les quatre principales couches de mouvement ont

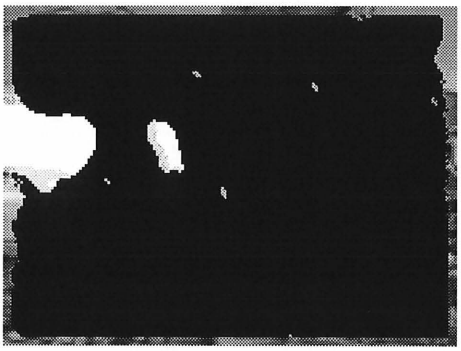


a. Couche 1 : arbre



b. Couche 2 : fleurs

c. Couche 3 : maison



d. Couche 4 : ciel

Figure 2.16 – *Résultats expérimentaux au temps* $t = 9$

été extraites et correctement ordonnées. Cependant, nous remarquons la présence de certaines imprécisions principalement au niveau des frontières (e.g. le long des arbres en avant-plan et en arrière-plan). Tel que mentionné à la section précédente, ces erreurs sont prévisibles et sont principalement dues à la présence d'occlusions et au lissage impliqué dans l'estimation du flot optique. Par conséquent, ces résultats sont satisfaisants.

2.7.2 Influence des paramètres

Certaines questions persistent en ce qui a trait aux valeurs des paramètres λ et K devant être utilisées lors de l'estimation du flot optique. Pour sa part, λ pondère l'erreur reliée à la contrainte du flot optique (équation (2.3)) par rapport au lissage. Plus λ est grand, moins la solution finale risque de comporter des valeurs aberrantes. Cependant, un lissage fort atténue les variations brusques. Il occasionne donc des erreurs principalement au niveau de la localisation des frontières entre les différentes couches (se référer à la section 2.6).

Le paramètre K détermine le nombre maximal d'itérations. Conformément au principe des algorithmes de raffinement successif, le calcul du flot optique tend à se stabiliser lorsque K est grand. Au-delà d'un certain seuil, la différence entre deux estimations successives (itérations k et $k+1$) n'est plus significative. Toutefois, l'utilisation d'une grande valeur accroît la complexité algorithmique. Le choix de K assure donc un compromis entre la rapidité d'exécution et la précision du flot optique estimé.

2.8 Conclusion

L'ordre de profondeur relatif des objets peut être estimé à partir du mouvement apparent présent dans une séquence vidéo. À cet effet, nous avons proposé une approche simple et rapide qui se divise en deux étapes. Dans un premier temps, le flot optique est estimé

à partir d'une version légèrement modifiée de l'algorithme de Horn et Schunk. Par la suite, les images sont segmentées à partir d'un critère d'homogénéité appliqué au module des vecteurs de mouvement. Le principe de cette approche repose sur une estimation robuste du mouvement. Or, cela représente un problème complexe. À ce jour, plusieurs algorithmes ont été suggérés pour tenter de le solutionner. Toutefois, aucun de ceux que nous avons testés n'a réussi à résoudre entièrement les problèmes d'ouvertures, de solutions multiples et d'occlusions s'y rattachant. L'étude et les tests comparatifs que nous avons effectués dans un contexte de segmentation en couches des images l'ont d'ailleurs confirmé. Notons que notre comparaison est qualitative. Une évaluation quantitative dans un même contexte reste à faire. Bien que les estimations de mouvement fournies ne soient pas parfaites, les résultats de l'algorithme de segmentation ont tout de même montré qu'elles peuvent être utilisées dans un processus de segmentation en couches des images. Pour arriver à de meilleurs résultats, il faudra cependant employer des outils et des techniques additionnels permettant la détection des occlusions. De tels outils permettront de marquer les zones d'occlusions et d'éviter de calculer le flot optique en ces endroits.

CHAPITRE 3

ESTIMATION DU RELIEF À PARTIR DU FLOU

3.1 Introduction

Dans le domaine de la vision artificielle, l'information de profondeur nous permet de comprendre la relation tridimensionnelle qui existe entre les différents objets de l'espace. Elle est utilisée dans de nombreux champs d'application, tels que la robotique, la médecine, la télédétection et le contrôle de qualité. Les approches pour estimer la profondeur diffèrent de par le nombre d'images utilisées (une ou plusieurs prises de différents points de vue et/ou à différents moments), le système de formation de l'image (actif ou passif) et les caractéristiques de l'image (niveaux de gris, contours, régions, textures, mouvement, perspective, flou, etc.). Les diverses approches reposent sur des hypothèses variées et fournissent des résultats adéquats dans des contextes différents. Plus précisément, nous nous intéressons au calcul de la profondeur à partir des variations de flou. En d'autres termes, il s'agit de calculer la distance entre une surface visible d'un objet et la lentille

mince de la caméra optique à partir de la quantité de flou. À cet effet, nous considérons une caméra positionnée à un endroit donné de l'espace 3D et qui génère une ou plusieurs images à niveaux de gris d'une scène 3D statique. Sous l'hypothèse d'une projection perspective, l'intensité de l'image d'un pixel donné (x,y) dépend de la distance z entre la lentille de la caméra et le point correspondant de la scène (figure 3.1). Dans le cas

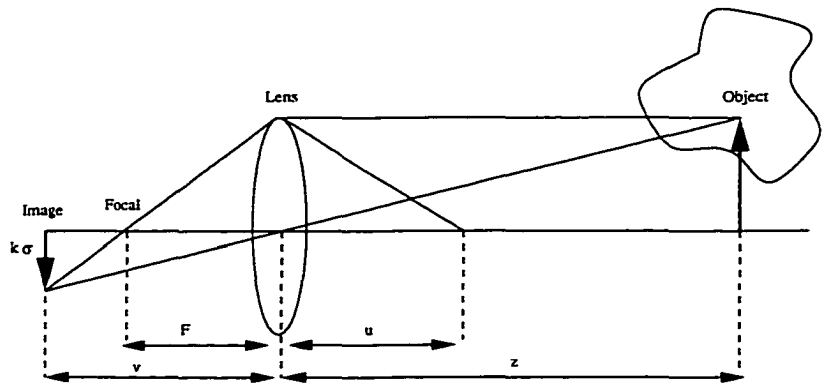


Figure 3.1 – *Système de formation de l'image pour une lentille mince*

d'une lentille mince, l'image est formée par la convolution de l'image idéalement projetée et $g(x,y,\sigma(x,y))$, la fonction d'étalement d'un point (*point spread function* (PSF)) de la caméra. $\sigma(x,y)$ représente le paramètre de flou du pixel (x,y). Dans la suite de ce chapitre, nous supposerons que $\sigma(x,y)$ est constant à l'intérieur d'une fenêtre donnée, c'est-à-dire $\sigma(x,y) = \sigma$. Cette hypothèse n'est généralement pas réaliste puisque le flou σ d'un pixel donné est en fonction de l'emplacement du point correspondant dans la scène. En fait, lorsque les points d'un objet ne sont pas tous à la même distance de la lentille, le flou dans l'image n'est pas constant. Cependant, sans cette hypothèse, nous sommes confrontés à un système qui n'est pas invariant par translation, et par conséquent difficile à résoudre. Cela justifie d'ailleurs la présence de cette hypothèse dans la majorité des travaux précédents. Il existe de nombreuses fonctions d'approximation de la PSF d'une caméra. De plus amples informations sont disponibles dans [59]. Pour notre part, nous nous intéressons au cas le plus commun, c'est-à-dire lorsque que l'approximation de la

PSF consiste en une fonction gaussienne :

$$g_\sigma(x,y) = \frac{1}{\pi\sigma^2} e^{-\frac{x^2+y^2}{\sigma^2}}. \tag{3.1}$$

La relation entre la distance z (figure 3.1) et le flou σ est donnée par [39] :

$$z = \begin{cases} \frac{Fv}{v-F-kf\sigma} & \text{if } z > u \\ \frac{Fv}{v-F+kf\sigma} & \text{if } z < u, \end{cases} \tag{3.2}$$

où u est la distance entre la lentille et la position du focus parfait, v la distance entre la lentille et le plan image, F la distance focale, f l'ouverture (*f-number*) et k le coefficient de proportionnalité entre le rayon du cercle de flou et σ. Cette équation montre que le calcul de la profondeur requiert une estimation préalable de f, F, v, k et σ. D'une part, les paramètres intrinsèques de la caméra (f, F, v et k) sont indépendants de la position du pixel (x,y). Ils peuvent ainsi être calculés par un processus de calibration [40, 61, 18]. D'autre part, le flou σ d'un pixel (x,y) est mesuré en calculant la fonction PSF de ce dernier.

Ce chapitre présente un algorithme permettant de calculer la profondeur à partir de la différence de flou entre deux images acquises à partir d'une même caméra dont les paramètres intrinsèques ont été modifiés. Pour ce faire, nous supposons que le système de formation d'image est passif. L'algorithme proposé implique l'approximation des images par une base du polynôme d'Hermite. Nous démontrerons que tout coefficient du polynôme d'Hermite, calculé à partir de l'image la plus floue, peut être exprimé en fonction des dérivées partielles de la seconde image et de la différence de flou. Il est ainsi possible d'estimer directement cette dernière en résolvant un système d'équations. Tous ces calculs sont locaux et sont effectués dans le domaine spatial. De plus, ils ne reposent pas sur un ou plusieurs modèles donnés d'images. Ils fournissent donc une estimation dense du flou, et par conséquent de la profondeur.

La section suivante donne un aperçu des approches existantes. Afin de faciliter la compréhension de cet algorithme, la section 3.3 décrit d'abord les règles d'estimation du flou

pour les images 1D. Par la suite, le modèle mathématique 1D est étendu aux images 2D. L'algorithme résultant est décrit à la section 3.5. La section 3.6 détaille le comportement de cet algorithme, présente les résultats expérimentaux et décrit l'influence du choix des différents paramètres.

3.2 Travaux connexes

Les premières recherches effectuées dans le but de valider l'utilisation des informations de flou en tant qu'indice de profondeur furent menées par Pentland [37] . Il démontra que deux images obtenues à partir d'ouvertures différentes fournissent des informations de profondeur. Depuis, plusieurs autres méthodes d'estimation du flou furent proposées. Celles-ci diffèrent principalement au niveau de quatre aspects. Premièrement, le nombre d'images que requiert l'algorithme (une, deux ou même plus). Lorsque la structure de l'image est connue, une image s'avère suffisante. Dans le cas contraire, une seule image d'une scène inconnue ne contient pas suffisamment d'information pour déterminer la fonction PSF de chaque pixel. Dans un tel cas, au moins deux images de la même scène obtenues en changeant un ou plusieurs paramètres de la caméra doivent être utilisées. Deuxièmement, les algorithmes proposés opèrent soit dans le domaine fréquentiel ou le domaine spatial. Troisièmement, ils produisent une estimation dense de la profondeur ou bien ils ne l'estiment que pour le voisinage des contours. Finalement, les approches les plus communes reposent sur le filtrage inverse, l'estimation de la profondeur à partir des contours ou la transformée en S. Voici un aperçu de chacune d'elles.

Le filtrage inverse fut d'abord utilisé par Pentland [37, 39] dans le domaine fréquentiel. En supposant que la fonction PSF est une gaussienne, considérons deux images telles que $I_c(x,y) = (I * g_{\sigma_c})(x,y)$ et $I_b(x,y) = (I * g_{\sigma_b})(x,y)$, où $\sigma_b > \sigma_c$. L'estimation de la différence de flou s'effectue en deux étapes. D'abord, il divise la transformée de Fourier des

deux images, c'est-à-dire $\mathcal{I}_b(u,v)/\mathcal{I}_c(u,v) = exp(-(u^2+v^2)(\sigma_b^2-\sigma_c^2)/4)$, où $\mathcal{I}(u,v)$ est la transformée de Fourier de $I(x,y)$. Ensuite, il effectue une régression linéaire sur u^2+v^2 de l'équation $log(\mathcal{I}_b(u,v)/\mathcal{I}_c(u,v)) = -(u^2+v^2)(\sigma_b^2-\sigma_c^2)/4$. Comme alternative à l'utilisation du domaine de Fourier et de la régression linéaire, Pentland *et al.* [38] proposent l'emploi du théorème de Parseval afin d'estimer la quantité $log(\mathcal{I}_b(u,v)/\mathcal{I}_c(u,v))$. Dans un même ordre d'idées, Horii [18] suggère l'utilisation du Laplacien de la gaussienne à deux échelles différentes pour estimer u^2+v^2 dans le but d'éviter l'emploi de la régression linéaire. Pour leur part, Xiong et Shafer [61] proposent une technique itérative permettant de déterminer la différence $\sigma_b^2-\sigma_c^2$. À la n^e itération, $\Delta_k = \sigma_b^2-\sigma_c^2$ est évaluée à l'aide de la formule $log(\mathcal{I}_b(u,v)/\mathcal{I}_c(u,v))$. L'image $I_c(x,y)$ est ensuite convoluée avec la différence de flou cumulative ($\sum_k \Delta_k$), puis comparée avec l'image $I_b(x,y)$. Ce processus se termine lorsque la ressemblance entre les deux images est maximale. Toutes ces techniques souffrent toutefois des limites de précision reliées à l'utilisation du filtrage inverse, tel que démontré par Ens et Lawrence [12]. Pour contourner ce problème, ils introduisent une contrainte de régularisation. Plus précisément, le problème de l'estimation de la différence de flou équivaut à la minimisation de la différence quadratique entre les deux images $I_c(x,y)$ et $I_b(x,y)$ en présence d'une contrainte de lissage. Finalement, Rajagopalan et Chaudhuri [45] proposent deux approches. Dans la première, l'image est d'abord divisée en sous-images. Pour une sous-image donnée, le flou est considéré constant. Par la suite, la différence de flou de chaque sous-image est estimée par la minimisation d'une différence quadratique pondérée entre les amplitudes de la transformée de Fourier de $I_c(x,y)$ et de $I_b(x,y)$. La seconde approche est basée sur la distribution de Wigner qui offre un filtrage adaptatif. Dans la majorité des cas, les algorithmes proposés reposent sur l'utilisation de la transformée de Fourier ou le théorème de Parseval. Or, la transformée de Fourier fournit une information globale, c'est-à-dire qu'une seule valeur de flou peut être calculée pour l'image entière. Afin d'obtenir une estimation plus dense, certaines techniques emploient des méthodes de calculs locales basées notamment sur la transformée de Fourier

court terme [61], les filtres de Gabor [17] et les filtres moments [62].

La profondeur peut également être déduite à partir des contours. Rappelons que si la structure de l'image est connue, une seule image contient suffisamment d'information pour estimer le flou, et par conséquent la profondeur. En ce sens, Pentland [39] considère l'estimation du flou au niveau des contours de type marche verticaux en supposant que la fonction PSF est une gaussienne. Sachant que la réponse du Laplacien de la gaussienne d'une marche est égale à $-xexp(-x^2/2\sigma^2)/\sqrt{2\pi}\sigma^3$, Pentland calcule le flou σ à partir du logarithme de cette dernière et d'une régression linéaire. Comme alternative à la régression linéaire, Lai *et al.* [24] approximent les contours linéaires de type marche à l'aide des moindres carrés. L'algorithme résultant est itératif et moins sensible au bruit présent dans l'image. Pour leur part, Subbarao et Gurumoorty [50] utilisent le concept de la fonction d'étalement de la ligne $\theta(x) = o^{'}(x)/\int_{-\infty}^{+\infty} o(x)dx$, où $o(x)$ est la convolution de la PSF et d'une marche verticale, et $o^{'}(x)$ la première dérivée de $o(x)$. Le flou est défini comme étant la racine carrée du second moment central de $\theta(x)$.

La transformée en S, proposée par Subbaroa et Surya [51], permet l'estimation d'une carte de profondeurs dense à partir de deux images $I_c(x,y)$ et $I_b(x,y)$. Les auteurs approchent l'image à l'aide d'un polynôme cubique $p(x,y) = \sum_{m=0}^{3}\sum_{n=0}^{3-m} a_{n,m}x^m y^n$. Ils démontrent que $I_c(x,y) = I_b(x,y) - \beta^2\nabla I_b(x,y)/4$, où ∇ est l'opérateur Laplacien et β la différence de flou; $\beta = \sqrt{\sigma_b^2 - \sigma_c^2}$. L'estimation de β, et par conséquent de la profondeur, est directe et tous les calculs requis sont effectués dans le domaine spatial. Cependant, certaines restrictions s'appliquent à cet algorithme. En fait, l'approximation de $I_c(x,y)$ est effectuée à l'aide d'un polynôme cubique $p(x,y)$. Étant donné que $g(x,y)$ est symétrique en x et en y, tous les termes du polynôme résultant sont nuls, sauf ceux de degrés (0,0), (0,2) et (2,0). Sachant que les termes (0,2) et (2,0) définissent le Laplacien, lorsque ce dernier est nul ou faible, par exemple aux points de contours ou de jonctions, la formule proposée par Subbarao et Surya ne permet pas d'estimer le flou. Or, celui-ci

peut très bien avoir un sens en ces pixels de l'image. Par exemple, pour les contours ou jonctions de marches résultant d'un changement d'illumination ou de réflectance sans variation de profondeur (exemple à la figure 3.2), le flou est bel et bien défini. Notre travail

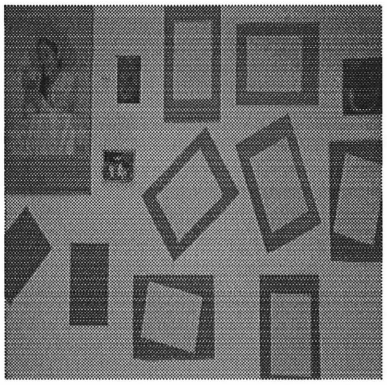


Figure 3.2 – *Image formée de posters placés contre un mur*

constitue une généralisation de cette méthode. En fait, tel que démontré par Ziou [63], l'utilisation des polynômes d'Hermite nous permet d'éliminer les restrictions énoncées précédemment. Il en découle un système composé de deux équations au niveau 1D et de quatre équations dans le cas des images 2D. L'algorithme qu'il suggère permet donc d'obtenir une estimation du flou plus lisse, plus dense et plus précise que celle fournie par l'algorithme de Subbarao et Surya. À partir de cette estimation, il est donc possible d'estimer de manière dense et précise la profondeur [65]. Un tel algorithme ne nécessite donc pas de processus itératif, notamment employé par Xiong et Shafer, Ens et Lawrence, Lu *et al.*, ainsi que Rajagopalan et Chaudhuri. Mentionnons également que contrairement aux algorithmes utilisant le filtrage inverse dans le domaine de Fourier (l'article de Ens et Lawrence révèle quelques problèmes sous-jacents), tous les calculs sont effectués dans le domaine spatial.

3.3 Estimation du flou pour un signal 1D

Cette section présente le développement mathématique sur lequel repose l'approche d'estimation du flou dans le cas d'un signal 1D. La généralisation de cette approche sera effectuée à la section suivante. Considérons $I_c(x)$ et $I_b(x)$, deux images d'une même scène obtenues en modifiant les paramètres intrinsèques d'une ou plusieurs caméras (se référer à la figure 3.1). Supposons que $I_b(x)$ est plus floue que $I_c(x)$. Notons qu'une telle supposition n'a aucune conséquence, puisqu'il est possible de déterminer quelle image est la plus floue. Par exemple, ceci peut être fait en identifiant l'image ayant la plus faible variance de niveaux de gris. Lorsque la fonction PSF est une gaussienne, les images sont formées selon les formules suivantes :

$$I_c(x) = (I * g_{\sigma_c})(x) \qquad \text{et} \qquad I_b(x) = (I * g_{\sigma_b})(x) \tag{3.3}$$

où $g_\sigma(x)$ est une gaussienne de variance σ^2, $I(x)$ l'image en focus et $*$ la convolution. La fonction gaussienne satisfait $g_{\sqrt{\sigma_c^2+\beta^2}}(x) = (g_{\sigma_c} * g_\beta)(x)$. Par conséquent, nous pouvons déduire la relation suivante entre $I_b(x)$ et $I_c(x)$:

$$I_b(x) = (I_c * g_\beta)(x) \tag{3.4}$$

Le paramètre $\beta = \sqrt{\sigma_b^2 - \sigma_c^2}$ correspond à la différence de flou entre les deux images. La relation entre β et z peut donc être déduite. À titre d'exemple, considérons une image $I_c(x)$ obtenue à partir d'une caméra dont les valeurs des paramètres sont v, F, f, et σ. L'image $I_b(x)$ provient de la même caméra dont la valeur du paramètre v a subit une modification δv (équation (3.2)), ce qui entraîne un changement d'amplitude β^2 du flou σ^2. La profondeur z est donnée par (se référer à l'Annexe C) :

$$\frac{1}{z} = \frac{1}{F} - \frac{1}{2v + \delta v}\left(1 + \sqrt{1 + \frac{f^2(2v + \delta v)}{F^2 \delta v} k^2 \beta^2}\right). \tag{3.5}$$

Cette équation démontre que la profondeur peut être calculée à partir des paramètres intrinsèques de la caméra et de la différence de flou β entre les deux images. Rappelons que

les paramètres intrinsèques peuvent être obtenus par calibration. Il ne reste donc qu'à déterminer β. À cette fin, nous avons développé une technique de décomposition de l'image appelée la transformée en polynômes d'Hermite. Celle-ci consiste à faire une approximation locale de l'image, à l'intérieur d'une fenêtre gaussienne $g_\sigma(x)$, par une base des polynômes d'Hermite. Plus précisément, nous considérons la fenêtre unidimensionnelle gaussienne :

$$g(x) = \frac{1}{\sqrt{\pi}\sigma} e^{-\frac{x^2}{\sigma^2}}. \tag{3.6}$$

Le n^e polynôme d'Hermite $H_n(x)$ est défini par :

$$H_n(x) = (-1)^n e^{x^2} \frac{d^n}{dx^n} e^{-x^2}. \tag{3.7}$$

Afin de simplifier le processus, plutôt que d'utiliser la base polynomiale $H_n(x)$, nous considérerons la séquence de polynômes suivante :

$$\{P_n(x) = H_n(\frac{x}{\sigma})\}. \tag{3.8}$$

Cette séquence est orthogonale pour une fonction $g_\sigma(x)$ définie sur l'intervalle $]-\infty, +\infty[$ (voir Annexe A). Le choix des polynômes d'Hermite est relié au fait que ceux-ci sont associés à la fonction gaussienne. Ainsi, ils possèdent des propriétés intéressantes et leur usage dans le domaine du traitement d'images est sans cesse grandissant [13, 26, 25, 21, 47]. À titre d'exemple, ils sont orthogonaux au sein d'une fenêtre gaussienne (voir Annexe A). Mentionnons que l'utilisation de ce type de fenêtre se justifie de plusieurs manières. D'abord, la fonction gaussienne est largement utilisée en traitement d'images. Ses propriétés mathématiques sont clairement définies et bien connues. Elle est, entre autres, séparable et permet ainsi la décomposition d'un problème multidimensionnel en problème unidimensionnel. Cela permettra d'ailleurs d'étendre facilement notre algorithme 1D afin qu'il fonctionne pour des dimensions multiples. Ensuite, de nombreuses méthodes ont été

proposées pour une implantation efficace de la gaussienne et de ses dérivées [48]. Finalement, nous démontrerons dans la suite de ce chapitre que la calcul du n^e coefficient du polynôme d'Hermite correspond à la convolution de l'image et de la n^e dérivée de la fonction gaussienne (à un facteur constant $(-1)^n\sigma^n$ près). Selon ce modèle mathématique, la description complète d'une image implique que le processus d'approximation soit répété pour un nombre suffisant de pixel à l'intérieur de la fenêtre. En considérant le cas où l'espacement de l'échantillonnage T est équidistant, l'approximation de l'image $I_b(x)$ par la base polynomiale $\{P_n(x)\}$ à l'intérieure de la fenêtre $g_\sigma(x)$ est donnée par :

$$I_b(x) = \sum_{n=0}^{\infty} c_n(m) P_n(x-m), \tag{3.9}$$

où $c_n(m)$ est défini par :

$$c_n(m) = (I_b * h_n)(m). \tag{3.10}$$

Cette équation signifie que les coefficients $c_n(m)$ proviennent de la convolution de l'image $I_b(x)$ par le filtre :

$$h_n(x) = P_n(x) g_\sigma(x) = (-1)^n \frac{d^n}{d(\frac{x}{\sigma})^n} \frac{1}{\sqrt{\pi}\sigma} e^{-\frac{x^2}{\sigma^2}}. \tag{3.11}$$

Comme alternative à l'utilisation du modèle de reconstruction de $I_b(x)$ de l'équation (3.9), qui est une fonction de m, nous proposons une description plus générale du modèle de l'image. Ce dernier nécessite que le processus d'approximation soit répété pour $2K+1$ pixels appartenant à la fenêtre. De plus, au lieu d'employer la fenêtre $g(x)$, nous proposons d'utiliser $w(x) = \sum_{k=-K}^{K} \sqrt{g_\sigma(x-kT)}$. En considérant que l'espacement de l'échantillonnage T est équidistant, l'approximation de l'image $I_b(x)$ par la base polynomiale $\{P_n(x)\}$ à l'intérieur de cette fenêtre centrée en x est :

$$I_b(x)w(x) = \sum_{k=-K}^{K} I_b(x)\sqrt{g_\sigma(x-kT)}. \tag{3.12}$$

En substituant $I_b(x)$ de la partie de droite par l'équation (3.9), nous obtenons :

$$I_b(x) = \sum_{n=0}^{\infty} \sum_{k=-K}^{K} c_n(kT) \frac{P_n(x - kT)\sqrt{g_\sigma(x - kT)}}{w(x)}. \tag{3.13}$$

De même, en introduisant l'équation (3.4) dans l'équation (3.10), les coefficients de l'image $I_b(x)$ deviennent :

$$c_n(kT) = (I_c * g_\beta * h_n)(kT) = (I_c * f_n)(kT), \tag{3.14}$$

où

$$f_n(x) = (g_\beta * h_n)(x) = (-1)^n \sigma^n \frac{d^n}{dx^n} \frac{1}{\sqrt{\pi}\alpha} e^{-\frac{x^2}{\alpha^2}} \tag{3.15}$$

et $\alpha^2 = \sigma^2 + \beta^2$. En d'autres termes, l'équation (3.14) peut s'écrire :

$$c_n(kT) = \int_{-\infty}^{+\infty} I_c(kT - x) f_n(x) dx. \tag{3.16}$$

Les équations (3.10) et (3.16) illustrent la relation qui existe entre les images I_b et I_c. Étant donné que seule la fonction $f_n(x)$ dépend de β, ces deux équations peuvent être utilisées pour estimer le flou. Cependant, cela revient à résoudre un système d'équations non linéaires. Il est donc nécessaire de simplifier cette relation afin d'en extraire une règle explicite de calcul du flou β. Considérons le développement en séries de Taylor de $I_c(kT - x)$ au point kT :

$$I_c(kT - x) = \sum_{p=0}^{+\infty} \frac{(-x)^p I_c^{(p)}(kT)}{p!}, \tag{3.17}$$

où $I_c^{(p)}(x)$ représente la p^e dérivée de $I_c(x)$. En combinant les équations (3.17) et (3.16), nous obtenons :

$$c_n(kT) = \sum_{p=0}^{+\infty} s_{n,p} I_c^{(p)}(kT), \tag{3.18}$$

où

$$s_{n,p} = \int_{-\infty}^{\infty} \frac{(-x)^p f_n(x)}{p!} dx. \tag{3.19}$$

Les équations (3.10) et (3.18) expriment la relation qui prévaut entre les images I_b et I_c. Le coefficient $s_{n,p}$ de cette dernière est proportionnel au p^e moment de $f_n(x)$ et est une fonction du flou β. Cela signifie qu'il est possible d'estimer β à partir de $I_c(x)$, $I_b(x)$ et $f_n(x)$. Toutefois, le coût associé au calcul direct des moments est très élevé. Pour contourner ce problème et permettre l'utilisation de techniques basées sur les moments dans les applications temps réel de la vision par ordinateur et du traitement d'images, diverses méthodes d'implantation ainsi que des architectures spécifiques furent proposées [44]. Dans ce qui suit, nous développons une procédure de calcul rapide pour $s_{n,p}$. À cette fin, nous distinguons les quatre situations suivantes :

- Lorsque $n = p = 0$, l'évaluation de l'équation (3.19) donne $s_{0,0} = 1$.
- Lorsque $p = 0$ et $n > 0$, $s_{n,0} = \int_{-\infty}^{+\infty} f_n(x)dx = 0$.
- Lorsque $n = 0$ et $p = 2r + 1$, la fonction $x^{2r+1} f_0(x)$ est impaire, alors $s_{0,2r+1} = \int_{-\infty}^{+\infty} x^{2r+1} f_0(x)dx = 0$. Lorsque $n = 0$ et $p = 2r$, $s_{0,2r}$ est calculé à l'aide de la formule suivante (Annexe B) :

$$s_{0,2r} = \frac{(2r-1)(2r-3)\cdots 3\alpha^{2r}}{(2r)!2^r}. \tag{3.20}$$

- Lorsque $n > 0$ et $p > 0$, à l'aide de la règle d'intégration par parties, l'équation (3.19) peut s'écrire :

$$s_{n,p} = -\sigma \frac{(-1)^{p-1}}{(p-1)!} \int_{-\infty}^{\infty} x^{p-1} f_{n-1}(x)dx, \tag{3.21}$$

ce qui est équivalent à :

$$s_{n,p} = -\sigma s_{n-1,p-1}. \tag{3.22}$$

Cette équation fournit une méthode récursive de calcul de $s_{n,p}$ pouvant être simplifiée :

$$s_{n,p} = \begin{cases} (-1)^p \sigma^p s_{n-p,0} & \text{si } n > p \\ (-1)^n \sigma^n s_{0,p-n} & \text{sinon.} \end{cases} \tag{3.23}$$

Selon les résultats précédents, $s_{n-p,0} = 0$ et $s_{0,p-n} = 0$ si $p-n$ est impair, c'est-à-dire si n ou p est impair. Sinon, $s_{0,p-n} = \frac{(p-n-1)(p-n-3)\cdots 3\alpha^{p-n}}{(p-n)!2^{(p-n)/2}}$.

Ces quatre cas peuvent être résumés par :

$$s_{n,p} = \begin{cases} \frac{(-1)^n \sigma^n}{(p-n)!2^{(p-n)/2}}(p-n-1)(p-n-3)\cdots 3\alpha^{p-n} & \text{si } 0 \leq n < p \text{ et } p-n \text{ est pair} \\ (-1)^n \sigma^n & \text{si } n = p \\ 0 & \text{sinon.} \end{cases} \tag{3.24}$$

Cette équation constitue une règle de calcul simple et efficace pour $s_{n,p}$. Elle permet ainsi la simplification des calculs de $c_n(kT)$ et l'approximation de $I_b(x)$. En utilisant les équations (3.10), (3.18) et (3.24), la relation entre les deux images $I_b(x)$ et $I_c(x)$ est désormais donnée par :

$$\begin{cases} c_{2r}(kT) = \sum_{p\geq r}^{+\infty} s_{2r,2p} I_c^{(2p)}(kT) = (I_b * h_{2r})(kT) \\ c_{2r+1}(kT) = \sum_{p\geq r}^{+\infty} s_{2r+1,2p+1} I_c^{(2p+1)}(kT) = (I_b * h_{2r+1})(kT). \end{cases} \tag{3.25}$$

L'équation de reconstruction de l'image $I_b(x)$ (éq. (3.13)) devient donc :

$$I_b(x) = \sum_{n=0}^{+\infty} \sum_{p\geq n}^{+\infty} \sum_{k=-K}^{K} (s_{2n,2p} I_c^{(2p)}(kT) P_{2n}(x-kT) + s_{2n+1,2p+1} I_c^{(2p+1)}(kT) P_{2n+1}(x-kT)) \frac{\sqrt{g_\sigma(x-kT)}}{w(x)} \tag{3.26}$$

Il est maintenant possible de calculer la différence de flou. À titre d'exemple, considérons l'utilisation des coefficients c_0 et c_1 dans l'équation (3.25). Dans le cas de c_0, nous supposons que $\sum_{p>1} s_{0,2p} I_c^{(2p)}(kT) = 0$, tandis que pour c_1, $\sum_{p>1}^{+\infty} s_{1,2p+1} I_c^{(2p+1)}(kT) = 0$.

Selon l'équation (3.24), $s_{0,0} = 1$, $s_{1,1} = -\sigma$, $s_{0,2} = \frac{\alpha^2}{4}$, et $s_{1,3} = -\sigma\frac{\alpha^2}{4}$. L'équation (3.25) devient donc :

$$\begin{cases} c_0(kT) = I_c(kT) + \frac{\alpha^2}{4} I_c''(kT) = (I_b * h_0)(kT) \\ c_1(kT) = -\sigma I_c'(kT) - \sigma\frac{\alpha^2}{4} I_c'''(kT) = (I_b * h_1)(kT). \end{cases} \tag{3.27}$$

où $h_0(kT)$ et $h_1(kT)$ correspondent respectivement à un filtre gaussien de variance $\sigma^2/2$ et à sa première dérivée multipliée par un facteur constant $(-\sigma)$ (équation (3.11)). En résolvant ces deux équations, nous obtenons :

$$\beta = \sqrt{4\frac{(I_b * h_0)(kT) - I_c(kT)}{I_c''(kT)} - \sigma^2} \tag{3.28}$$

ou

$$\beta = \sqrt{4\frac{(I_b * h_1)(kT) + \sigma I_c'(kT)}{-\sigma I_c'''(kT)} - \sigma^2} \tag{3.29}$$

3.4 Estimation du flou pour une image 2D

L'extension de l'approche unidimensionnelle aux images multidimensionnelles est directe. En 2D, un polynôme d'Hermite peut être créé à partir du produit tensoriel entre deux ensembles de polynômes d'Hermite unidimensionnels, c'est-à-dire $\{P_0(x)P_0(y), \cdots, P_n(x)P_m(y), \cdots\}$, où n est le degré de x et m celui de y. Étant donné qu'une fenêtre gaussienne $g_\sigma(x,y)$ est séparable, on peut aisément démontrer que $\{P_{n,m}(x,y)\}$ est orthogonal à l'intérieur de $g_\sigma(x,y)$. Supposons que T_x et T_y représentent respectivement l'espacement de l'échantillonnage selon l'axe des x et des y au sein de la fenêtre. La décomposition locale de l'image $I_b(x,y)$ en polynômes devient :

$$I_b(x,y) = \sum_{n=0}^{\infty}\sum_{m=0}^{+\infty}\sum_{k=-K}^{K}\sum_{l=-L}^{L} c_{n,m}(kT_x,lT_y)P_{n,m}(x - kT_x, y - lT_y)\frac{\sqrt{g_\sigma(x - kT_x, y - lT_y))}}{w(x,y)}, \tag{3.30}$$

où $(2K+1) \times (2L+1)$ correspond à la dimension de la matrice d'échantillonnage et $c_{n,m}(kT_x,lT_y)$ est défini par :

$$c_{n,m}(kT_x,lT_y) = (I_b * h_{n,m})(kT_x,lT_y), \tag{3.31}$$

où $h_{n,m}(x,y) = P_{n,m}(x,y)g_\sigma(x,y)$. Puisque $P_{n,m}(x,y)$ et $g_\sigma(x,y)$ sont toutes deux séparables, une transformation directe, telle que celle présentée dans le cas 1D, donne :

$$c_{n,m}(kT_x,lT_y) = (I_c * f_{n,m})(kT_x,lT_y), \tag{3.32}$$

où $f_{n,m}(x,y)$ est séparable :

$$f_{n,m}(x,y) = (-1)^{n+m}\frac{d^n}{d(\frac{x}{\sigma})^n}\frac{1}{\sqrt{\pi}\alpha}e^{-\frac{x^2}{\alpha^2}}\frac{d^m}{d(\frac{y}{\sigma})^m}\frac{1}{\sqrt{\pi}\alpha}e^{-\frac{y^2}{\alpha^2}}. \tag{3.33}$$

En développant $I_c(kT_x - x,lT_y - y)$ en séries de Taylor au point (kT_x,lT_y), nous obtenons :

$$I_c(kT_x - x,lT_y - y) = \sum_{p=0}^{+\infty}\sum_{q=0}^{+\infty}\frac{(-x)^p(-y)^q I_c^{(p)(q)}(kT_x,lT_y)}{p!q!}, \tag{3.34}$$

où $I_c^{(p)(q)}(x,y) = \frac{\partial^{p+q} I_c(x,y)}{\partial x^p \partial y^q}$. En substituant cette dernière équation dans (3.32), il en résulte :

$$c_{n,m}(kT_x,lT_y) = \sum_{p=0}^{+\infty}\sum_{q=0}^{+\infty} s_{n,p,m,q} I_c^{(p)(q)}(kT_x,lT_y), \tag{3.35}$$

où $s_{n,p,m,q}$ est défini par :

$$s_{n,p,m,q} = \int_{-\infty}^{+\infty}\int_{-\infty}^{\infty}\frac{(-x)^p(-y)^q f_{n,m}(x,y)}{p!q!}dxdy. \tag{3.36}$$

Étant donné que $f_{n,m}(x,y)$ est séparable, $s_{n,p,m,q} = s_{n,p}s_{m,q}$, c'est-à-dire :

$$s_{n,p,m,q} = \int_{-\infty}^{\infty}\frac{(-x)^p f_n(x)}{p!}dx\int_{-\infty}^{\infty}\frac{(-y)^q f_m(y)}{q!}dy. \tag{3.37}$$

Un calcul rapide de $s_{n,p,m,q}$ est possible à partir de l'équation (3.24). Les coefficients $c_{n,m}(kT_x,lT_y)$ de l'équation (3.35) deviennent donc :

$$c_{n,m}(kT_x,lT_y) = \sum_{p\geq n}^{+\infty}\sum_{q\geq m}^{+\infty} s_{n,p,m,q} I_c^{(p)(q)}(kT_x,lT_y), \tag{3.38}$$

et peuvent être simplifiés :

$$\begin{cases} c_{2r,2t}(kT_x,lT_y) = \sum_{p\geq r}^{+\infty}\sum_{q\geq t}^{+\infty} s_{2r,2p}s_{2t,2q}I_c^{(2p)(2q)}(kT_x,lT_y) \\ c_{2r,2t+1}(kT_x,lT_y) = \sum_{p\geq r}^{+\infty}\sum_{q\geq t}^{+\infty} s_{2r,2p}s_{2t+1,2q+1}I_c^{(2p)(2q+1)}(kT_x,lT_y) \\ c_{2r+1,2t}(kT_x,lT_y) = \sum_{p\geq r}^{+\infty}\sum_{q\geq t}^{+\infty} s_{2r+1,2p+1}s_{2t,2q}I_c^{(2p+1)(2q)}(kT_x,lT_y) \\ c_{2r+1,2t+1}(kT_x,lT_y) = \sum_{p\geq r}^{+\infty}\sum_{q\geq t}^{+\infty} s_{2r+1,2p+1}s_{2t+1,2q+1}I_c^{(2p+1)(2q+1)}(kT_x,lT_y). \end{cases} \tag{3.39}$$

Par conséquent, l'équation de reconstruction de l'image $I_b(x,y)$ (3.30) devient :

$$\begin{aligned} I_b(x,y) \;=\; & \sum_{n0}^{+\infty}\sum_{m=0}^{+\infty}\sum_{p\geq n}^{+\infty}\sum_{q\geq m}^{+\infty}\sum_{k=-K}^{K}\sum_{l=-L}^{L} \big(s_{2n,2p}s_{2m,2q}I_c^{(2p)(2q)}(kT_x,lT_y) \\ & P_{2n}(x-kT_x)P_{2m}(y-lT_y) + \\ & s_{2n,2p}s_{2m+1,2q+1}I_c^{(2p)(2q+1)}(kT_x,lT_y)P_{2n}(x-kT_x)P_{2m+1}(y-lT_y) + \\ & s_{2n+1,2p+1}s_{2m,2q}I_c^{(2p+1)(2q)}(kT_x,lT_y)P_{2n+1}(x-kT_x)P_{2m}(y-lT_y) + \\ & s_{2n+1,2p+1}s_{2m+1,2q+1}I_c^{(2p+1)(2q+1)}(kT_x,lT_y)P_{2n+1}(x-kT_x)P_{2m+1}(y-lT_y)\big) \\ & \frac{\sqrt{g_\sigma(x-kT_x,y-lT_y)}}{w(x,y)}. \end{aligned} \tag{3.40}$$

Il est maintenant possible de calculer la différence de flou entre deux images 2D. À titre d'exemple, considérons l'utilisation des coefficients $c_{0,0}$, $c_{0,1}$, et $c_{1,0}$ dans l'équation (3.41). Dans le cas de $c_{0,0}$, nous supposons que $\sum_{p\geq 2}^{+\infty}\sum_{q\geq 2}^{+\infty} s_{0,2p}s_{0,2q}I_c^{(2p)(2q)}(kT_x,lT_y) = 0$, pour $c_{0,1}$, $\sum_{p\geq 2}^{+\infty}\sum_{q\geq 1}^{+\infty} s_{0,2p}s_{1,2q}I_c^{(2p)(2q)}(kT_x,lT_y) = 0$, tandis que pour $c_{1,0}$, $\sum_{p\geq 1}^{+\infty}\sum_{q\geq 2}^{+\infty} s_{1,2p}s_{0,2q}I_c^{(2p)(2q)}(kT_x,lT_y) = 0$. Selon les équations (3.24) et (3.31), les trois premières équations de (3.39) peuvent s'écrire :

$$\begin{aligned} c_{0,0}(kT_x,lT_y) &= I_c(kT_x,lT_y) + \tfrac{\alpha^2}{4}\nabla I_c(kT_x,lT_y) + (\tfrac{\alpha^2}{4})^2 I_c^{(2)(2)}(kT_x,lT_y) = (I_b * h_{0,0})(kT_x,lT_y) \\ c_{0,1}(kT_x,lT_y) &= -\sigma I_c^{(0)(1)}(kT_x,lT_y) - \sigma\tfrac{\alpha^2}{4} I_c^{(2)(1)}(kT_x,lT_y) = (I_b * h_{0,1})(kT_x,lT_y) \\ c_{1,0}(kT_x,lT_y) &= -\sigma I_c^{(1)(0)}(kT_x,lT_y) - \sigma\tfrac{\alpha^2}{4} I_c^{(1)(2)}(kT_x,lT_y) = (I_b * h_{1,0})(kT_x,lT_y) \end{aligned} \tag{3.41}$$

où ∇ est l'opérateur Laplacien. Mentionnons que la première équation dans (3.41) correspond à celle ayant été implantée par M. Subbarao et G. Surya [51] et dans laquelle

l'approximation de l'image est effectuée à l'aide de polynômes de degré inférieur ou égale à trois, c'est-à-dire $I_c^{(2)(2)}(x,y) = 0$. Il est donc clair que leur modèle d'estimation du flou constitue un cas particulier de celui que nous proposons (équation (3.39)).

3.5 Algorithme pour l'estimation du flou

Les développements mathématiques impliqués dans l'estimation de la différence de flou β (équation (3.4)) ayant été présentés dans les sections précédentes, nous devons maintenant spécifier l'algorithme qui en découle directement. Considérons une somme finie pour les équations (3.31) et (3.39) pour un n et m donnés (c'est-à-dire que nous négligeons les termes d'ordre supérieur) :

$$c_{n,m}(kT_x,lT_y) = (I_b * h_{n,m})(kT_x,lT_y), \tag{3.42}$$

$$\begin{cases} c_{2r,2t}(kT_x,lT_y) = \sum_{p\geq r}^{P}\sum_{q\geq t}^{Q} s_{2r,2p}s_{2t,2q}I_c^{(2p)(2q)}(kT_x,lT_y) \\ c_{2r,2t+1}(kT_x,lT_y) = \sum_{p\geq r}^{P}\sum_{q\geq t}^{Q} s_{2r,2p}s_{2t+1,2q+1}I_c^{(2p)(2q+1)}(kT_x,lT_y) \\ c_{2r+1,2t}(kT_x,lT_y) = \sum_{p\geq r}^{P}\sum_{q\geq t}^{Q} s_{2r+1,2p+1}s_{2t,2q}I_c^{(2p+1)(2q)}(kT_x,lT_y) \\ c_{2r+1,2t+1}(kT_x,lT_y) = \sum_{p\geq r}^{P}\sum_{q\geq t}^{Q} s_{2r+1,2p+1}s_{2t+1,2q+1}I_c^{(2p+1)(2q+1)}(kT_x,lT_y), \end{cases} \tag{3.43}$$

où $r = 0,1,\cdots,N$ et $t = 0,1,\cdots,M$. L'équation (3.42) et n'importe quelle équation de (3.43) forment un système de deux équations à deux inconnues : $c_{n,m}$ et $s_{n,p,m,q}$. $s_{n,p,m,q}$ est une fonction du flou β, lequel peut ainsi être estimé en résolvant ce système. Cependant, il y a quatre équations différentes dans (3.43). Nous estimons donc β à l'aide de l'équation (3.42) et de chacune des relations présentent dans l'équation (3.43). Pour chaque pixel (x,y), nous obtenons donc quatre valeurs de β, à partir desquelles nous choisissons celle qui minimise l'erreur quadratique calculée sur un voisinage de (x,y) :

$$e^2(x,y) = \sum_r \sum_s \left(I_b(x+r,y+s) - (I_c * g_\beta)(x+r,y+s)\right)^2. \tag{3.44}$$

Notons que les dérivées $I_c^{(p)(q)}(x,y)$, $p = 0,1,\cdots$, $q = 0,1,\cdots$ peuvent être estimées à l'aide de diverses techniques de différentiation numérique, telles que les masques de Sobel, les masques de Prewitt, les masques gaussiens, les ondelettes, l'approximation polynomiale, etc.

En résumé, soient σ, T_x, T_y, N, M, P, et Q, l'algorithme est composé de trois étapes :

1. Estimation des $c_{n,m}$ (équation (3.42)).
2. Estimation des quatres valeurs de β (équation (3.43)).
3. Sélection du meilleur β selon l'erreur quadratique obtenue (équation (3.44)).

Mentionnons que des définitions différentes de l'erreur quadratique peuvent également être utilisées.

3.6 Évaluation des performances

3.6.1 Comportement de l'algorithme

Jusqu'à maintenant, aucune supposition n'a été formulée concernant les images pour lesquelles notre algorithme fournit de bons résultats. Examinons donc son comportement en présence d'images constantes, de contours de type marche, de lignes et de jonctions de marches.

- Images constantes : les dérivées de l'image sont nulles et le système d'équations (3.42 et 3.43) est réduit à $c_{n,m}(kT_x,lT_y) = s_{n,0,m,0}I_c(kT_x,lT_y) = (I_b * h_{n,m})(kT_x,lT_y)$. Selon l'équation (3.24), $s_{n,0,m,0}$ vaut 1 si $n = m = 0$ et 0 dans le cas contraire. Ainsi, $c_{0,0}(kT_x,lT_y) = I_c(kT_x,lT_y) = (I_b * h_{0,0})(kT_x,lT_y)$. Cette équation ne permet donc pas le calcul du flou. En d'autres termes, les basses fréquences ne sont pas utilisées

par cet algorithme pour estimer le flou.

- Contours de type marche: considérons le modèle de contour $I(x,y) = \phi((x - kT_x)cos(\theta) + (y - lT_y)sin(\theta))$, où $\phi(x) = \int_{-\infty}^{x} exp(-t^2)dt$ et θ représente l'orientation du contour. $I^{(p)(q)}(kT_x,lT_y) = 0$, si p ou q est pair. Par conséquent, dans un tel cas, seulement la dernière équation de (3.43) peut servir à l'estimation du flou.
- Lignes: considérons le modèle de ligne $I(x,y) = e^{-((x-kT_x)*cos(\theta)+(y-lT_y)*sin(\theta))^2}$, où θ correspond à l'orientation de la ligne. $I^{(p)(q)}(kT_x,lT_y) = 0$, si p ou q est impair. Dans cette situation, seule la première équation de (3.43) peut donc être utilisée pour le calcul du flou.
- Jonctions de marche: prenons le modèle de jonction L suivant: $I(x,y) = \phi(x - kT_x)\phi(y - kT_y)$. $I^{(p)(q)}(kT_x,lT_y) = 0$, si p ou q est pair. Cela signifie que l'estimation du flou ne peut être effectuée qu'à partir de la dernière équation de (3.43). Un comportement similaire peut également être obtenu en présence de jonctions T: $I(x,y) = \phi(x - kT_x)(\phi(y - kT_y) + 1)$.

Examinons de plus près le comportement de la première équation de (3.41), qui est d'ailleurs implantée par Subbarao et Surya, pour laquelle l'approximation de l'image est effectuée à l'aide d'une base polynomiale de degré inférieur ou égale à trois. Au niveau des contours de type marche, elle ne permet pas de calculer le flou. Aux points de lignes, le Laplacien est non nul. Le flou peut donc être estimé. En ce qui concerne les modèles de jonctions linéaires de marches (l'intersection de n régions d'intensité constante comme les L, T, Y et X), le Laplacien de la gaussienne est nul au point d'intersection [54]. Il est ainsi impossible d'évaluer le flou en ce point. En conclusion, l'algorithme proposé par Subbarao et Surya ne peut servir à l'estimation du flou au niveau des contours et des jonctions de marches.

Des questions persistent en ce qui a trait à la signification du flou dans les images constantes, les lignes ainsi que les contours et coins de marches. Il faut d'abord savoir que

la convolution d'une image constante avec une distribution gaussienne normalisée (PSF) redonne l'image elle-même. Ainsi, pour les régions constantes de l'image, une estimation locale ne peut s'avérer suffisante pour le calcul du flou. Dans le cas des lignes, celles-ci proviennent généralement d'objets étroits placés devant un fond ou d'une illumination mutuelle entre des objets de la scène qui sont en contact. L'estimation de la profondeur d'objets minces a un sens et peut être utile. Cependant, dans une scène tridimensionnelle, la profondeur n'est pas définie au niveau d'un point d'occlusion. Il en découle que le flou ne l'est guère davantage. Lorsqu'un contour ou une jonction de marches résultent d'un tout autre phénomène (réflectance, illumination, etc.) n'impliquant aucun changement brusque de profondeur, le flou est toutefois bel et bien défini. La figure 3.2 illustre une telle situation.

3.6.2 Résultats expérimentaux

A) Estimation du flou

L'algorithme proposé a été testé à l'aide d'images 1D et 2D. Dans le cas 1D, nous avons implanté les équations (3.28) et (3.29). Pour ce test, les dérivées du signal sont calculées à partir de la convolution de ce dernier avec les dérivées appropriées de la fonction gaussienne. Les valeurs de l'échelle σ et de l'espacement T sont toutes deux égales à 1. La figure 3.3 présente un exemple d'estimation du flou dans le cas d'un signal 1D perturbé par un bruit blanc additif de moyenne zéro. Le signal de la figure 3.3b est plus flou que celui de la figure 3.3a. L'erreur quadratique moyenne (RMS) de l'estimation du flou en fonction de la variance du bruit est présentée dans la figure 3.3c. Plus précisément, les trois courbes correspondent à l'erreur découlant respectivement de l'utilisation des formules c_0 (éq. (3.28)) (points), c_1 (éq. 3.29) (tirets) et du modèle unifié (éq. 3.44) (gras). L'erreur quadratique résultant de l'utilisation séparée de c_0 ou c_1 est élevée. Cependant,

celle provenant du modèle unifié est plus faible et plus lisse. À titre d'exemple, lorsque la variance du bruit vaut 400, l'erreur RMS de c_0 est 2.29, celle de c_1 est 2.19, tandis que celle du modèle unifié est 1.38. Notons également que dans l'ensemble, l'erreur quadratique est élevée lorsque la variance du bruit tend vers zéro. Cependant, lorsque cette dernière est grande, son influence sur l'erreur RMS est faible. Ainsi, notre algorithme s'avère tout à fait approprié pour les images dont la variance des niveaux de gris est élevée, telles que les images texturées ou bruitées. Dans le cas d'images dont la variance est faible (e.g. régions constantes d'une image), il est recommandé d'utiliser des techniques globales. Finalement, les figures 3.3d, 3.3e et 3.3f présentent l'erreur quadratique moyenne (RMS) de l'estimation du flou en fonction de la variance du bruit et de la différence de flou entre les deux images. Elles sont respectivement associées aux équations c_0, c_1 et au modèle unifié. Tel qu'illustré, pour toutes les règles de calculs, l'erreur RMS est élevée lorsque la variance du bruit avoisine zéro. Cette erreur est également sensible à la différence de flou. Elle décroît en fonction de l'augmentation de cette dernière. En conclusion, ce test montre clairement que le modèle unifié est plus précis et moins sensible au bruit et à la différence de flou.

Dans le cas 2D, nous avons testé les équations de (3.41) avec deux images réelles (figures 3.4a et 3.4b) d'une maison jouet acquises par une caméra optique en modifiant l'ouverture et la durée d'exposition. Ces images sont de dimensions égales à 384×320 pixels et elles comportent 256 niveaux de gris. Elles nous ont été fournies par Y. Xiong du Robotics Institute de Carnegie Mellon University. La maison est à une distance d'environ deux mètres de la caméra. Elle a une hauteur de 4,5 pouces, une largeur de 2,5 pouces et une longueur de 3 pouces. Pour ce test, les dérivées de l'image sont calculées à partir de la convolution de cette dernière avec les dérivées appropriées de la fonction gaussienne. Les valeurs de l'échelle σ et des espacements T_x et T_y sont toutes trois égales à 1. Les figures 3.4c, 3.4d, et 3.4e présentent les estimations respectives du flou à partir de $c_{0,0}$,

$c_{0,1}$, et $c_{1,0}$ (éq. (3.41)). Les régions noires contenues dans la figure 3.4c correspondent à des endroits où le flou ne peut être calculé. Tel que montré dans les figures 3.4d et 3.4e, les utilisations de $c_{0,1}$ et de $c_{1,0}$ sont recommandées en présence d'éléments verticaux et horizontaux, respectivement. Le modèle unifié des trois estimations de flou (figures 3.4c, 3.4d, et 3.4e) est présenté à la figure 3.4f. Nous constatons que le nombre de régions noires présentes dans cette image est faible, ce qui signifie que l'estimation est plus dense.

B) Estimation de la profondeur

Notre algorithme d'estimation du flou a ensuite été validé dans un contexte d'estimation de la profondeur. En ce sens, les performances de la méthode d'estimation de la profondeur ont été testées sur de nombreuses images réelles. Ces images nous proviennent de A.N. Rajagopalan et S. Chaudhuri du Department of Electrical Engineering du Indian Institute of Technology-Bombay. Elles ont été acquises à l'aide d'une caméra optique dont la distance focale et l'ouverture (*f-number*) sont respectivement $F = 2.5$ cm et $f = 4$. Les figures 3.5a et 3.5b constituent la première paire d'images utilisée. Elles représentent une scène formée de deux plans verticaux positionnés obliquement par rapport à l'axe de la caméra et qui se croisent selon une droite verticale. Leur acquisition respective a été effectuée en ajustant la distance séparant la lentille du plan image à $v = 2.5532$ cm (figure 3.5a), puis à $v = 2.5714$ cm (figure 3.5b). L'intersection des deux plans se situe à 120 cm de la caméra, tandis que le point le plus près est à une distante de 105 cm. Dans un premier temps, le flou est calculé avec une échelle $\sigma = 1$ et des espacements $T_x = T_y = 1$. L'estimation résultante et l'équation (3.5) sont ensuite utilisées pour évaluer la profondeur des régions texturées, comme le montre la figure 3.5c. Nous observons sur ce graphique une distribution triangulaire de la profondeur, laquelle semble être adéquate. Effectivement, selon cette dernière, la distance du point le plus près est d'environ 107 cm et les plans se croisent à environ 119 cm.

La seconde paire d'images représente une scène contenant des variations rapides de profondeur (figures 3.6a et 3.6b). La distance séparant la caméra du point le plus loin est de 95 cm, tandis que celle entre la caméra et le point le plus proche est de 70 cm. Ces images ont été prises en utilisant $v = 2.5714$ cm (figure 3.6a) et $v = 2.5532$ cm (figure 3.6b). Le graphique de la profondeur des régions texturées en fonction de x et y est tracé dans la figure 3.6c. Nous constatons que les discontinuités de profondeur de type marche sont bel et bien apparentes. De plus, la distance séparant les points les plus loin de ceux qui sont les plus près est d'environ 24 cm. Ce qui est raisonnablement précis.

Les figures 3.7a et 3.7b correspondent à une scène contenant un objet plan texturé. Ces images ont été respectivement acquises pour $v = 2{,}5556$ cm et $v = 2{,}5714$ cm. Le point le plus loin de l'objet plan se situe à 125 cm de la caméra, tandis que le point le plus proche est à 115 cm. La profondeur estimée pour chacun des pixels appartenant à cet objet est montrée dans la figure 3.7c. En observant ce graphique, nous remarquons la nature plane de l'objet. Nous constatons également que la distance du point le plus loin et celle du plus près sont adéquates. En effet, l'erreur quadratique moyenne (RMS) de cette estimation est de 2,66 cm (2,21%), alors que sa densité est de 97,4%. À titre comparatif, nous avons également évalué la profondeur de cette scène en utilisant uniquement $c_{0,0}$ de l'équation (3.41)) (figure 3.8). Rappelons que $c_{0,0}$ correspond à la règle d'estimation du flou implantée par Subbarao et Surya [51]. L'erreur RMS et la densité associées à cette estimation sont respectivement 5,13 cm (4,22%) et 85,3%. Par conséquent, en comparant ces deux résultats, nous pouvons établir que le modèle unifié permet d'accroître la densité et la précision de la carte de profondeurs résultante.

3.6.3 Influence des paramètres

Certaines questions demeurent concernant les valeurs de T_x, T_y, N, M, P et Q devant être utilisées. Dans ce qui suit, nous discuterons du choix de ces paramètres selon l'ordre suivant : T_x et T_y, N et M, P et Q.

Le flou β est considéré comme étant localement constant. Supposons qu'il est calculé à l'intérieur d'une fenêtre W de dimensions $T_x \times T_y$. Ainsi, puisque W doit être petite, T_x et T_y doivent l'être également.

Selon l'équation (3.24), les paramètres N et M interviennent dans le calcul de $s_{n,p,m,q}$. Ils contrôlent le degré de flou β. Conformément à l'équation (3.41), la reconstruction de l'image est plus précise lorsque N et M sont grands. Or, cette précision n'affecte aucunement l'estimation du flou. Par conséquent, de petites valeurs pour N et M suffisent.

Les paramètres P et Q contrôlent à la fois le degré de flou β et l'ordre maximal des dérivées partielles de $I_c(x,y)$. Il n'existe aucune règle précise concernant les choix de ces deux paramètres. En fait, c'est un compromis entre quatre facteurs différents : la convergence des séries de l'équation (3.39), les hautes fréquences, les exigences de l'implantation et le problème de la différentiation numérique. La convergence des séries a été démontrée antérieurement (se référer à [64]). Concernant l'influence des hautes fréquences, considérons la dérivée d'ordre p et q d'une image I lissée à l'aide d'un filtre gaussien (équation (3.1)). Conformément au théorème de la convolution, $(I^{(p)(q)} * g_\sigma)(x,y) = (I * g_\sigma^{(p)(q)})(x,y)$. La transformée de Fourier de $g_\sigma^{(p)(q)}$ est $G(u,v) = (ju)^p(jv)^q e^{-\frac{\sigma^2}{2}(u^2+v^2)}$. En posant les dérivées partielles d'ordre un de $G(u,v)$ égalent à zéro et en résolvant les deux équations résultantes, nous obtenons $u = \frac{\mp\sqrt{p}}{\sigma}$ et $v = \frac{\mp\sqrt{q}}{\sigma}$. Les extrema de $G(u,v)$ correspondent aux positions $(\frac{\mp\sqrt{p}}{\sigma}, \frac{\mp\sqrt{q}}{\sigma})$. Cela signifie que, pour un σ donné, lorsque p et/ou q augmentent, les extrema s'éloignent de leur origine, et par conséquent, seules les hautes fréquences sont considérées. Ceci est également vrai lorsque σ décroît. Dans une image, les hautes

fréquences correspondent à des variations rapides du niveau de gris (bruit, contours, textures, etc.). Étant donné que notre algorithme utilise ce type d'information, il est donc important d'utiliser de grandes valeurs pour P et Q.

Pour leur part, les exigences reliées à l'implantation découlent du fait que les extrema des dérivées de la gaussienne s'éloignent de l'origine dans le domaine fréquentiel en fonction de l'ordre des dérivées (P et Q). En fait, la valeur des fréquences devant être considérées est une fonction de P et de Q. Cela signifie que l'augmentation de l'ordre de la dérivée de la gaussienne occasionne un accroissement de sa largeur dans le domaine spatial (les intervalles dans lesquels cette fonction est non nulle). Par conséquent, lorsque des masques de convolution sont employés pour l'implantation des dérivées de la gaussienne (une méthode courante), leurs dimensions augmentent avec l'ordre de ces dernières. Cela accroît la complexité algorithmique de la convolution et par le fait même de l'estimation du flou. De plus, les bords de l'image (régions qui ne sont pas traitées par la convolution) deviennent de plus en plus larges. En conclusion, les ordres des dérivées P et Q doivent être petits.

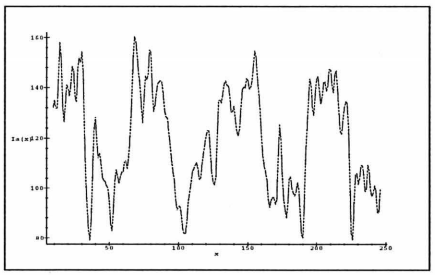
Finalement, concernant la différentiation numérique, il est bien connu que le calcul des dérivées d'images discrètes bruitées représente un problème mal posé. En effet, ces dérivées sont sensibles au bruit. Afin d'illustrer ceci, considérons le cas simple d'une image $s(x)$ modifiée par un bruit sinusoïdal additif d'amplitude a et de fréquence w, souvent appelé bruit cohérent ($i(x) = s(x) + a * sin(w * x)$). La perturbation de $i(x)$ est faible pour de petites valeurs de a. Cependant, la p^e dérivée de $i(x)$ peut être fort différente de $s^{(p)}(x)$ si $w >> 1$. Par conséquent, la perturbation des dérivées de l'image par un bruit sinusoïdal est élevée et tend à augmenter en fonction de leur ordre. Il est donc important d'utiliser de petites valeurs pour P et Q.

3.7 Conclusion

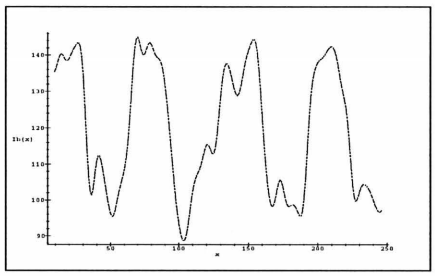
Il a récemment été démontré que l'estimation de la profondeur peut être effectuée à partir des variations de flou contenues dans une ou plusieurs images d'une même scène. À cet effet, nous avons proposé une approche permettant d'estimer le relief à partir de la différence de flou entre deux images acquises à partir d'une même caméra dont les paramètres intrinsèques ont été modifiés. Celle-ci se divise en trois étapes. Dans un premier temps, les images sont approchées par une base du polynôme d'Hermite. Nous avons démontré que tout coefficient d'un tel polynôme, calculé à partir de l'image la plus floue, peut être exprimé en fonction des dérivées partielles de la seconde image et de la différence de flou β. Dans un deuxième temps, l'unification et la résolution des systèmes d'équations ainsi créés (un pour chaque coefficient) permet d'obtenir différentes valeurs de β pour chaque pixel (x,y). Finalement, parmi toutes ces valeurs, celle minimisant l'erreur quadratique est retenue et utilisée dans le but de calculer la profondeur.

Les tests effectués ont clairement montré que la combinaison des différentes valeurs de flou à l'aide du modèle unifié (équation (3.41)) mène à l'obtention d'une estimation dense et précise du flou et de la profondeur. L'algorithme proposé est rapide et peut être implanté parallèlement. Les espacements d'échantillonnage T_x et T_y au sein de la fenêtre favorisent d'ailleurs une implantation efficace en réduisant les temps de calcul. De plus, tous les calculs impliqués sont locaux et sont effectués dans le domaine spatial.

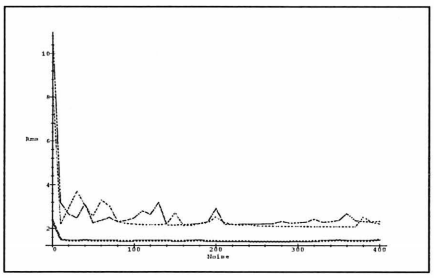
À ce jour, seul un cas particulier de notre algorithme a été implanté, c'est-à-dire l'utilisation de $c_{0,0}$, $c_{0,1}$ et $c_{1,0}$ de l'équation (3.41). Des implantations additionnelles concernant les règles présentées à l'équation (3.39) seront ultérieurement effectuées pour des valeurs différentes de r et t dans le but d'analyser leur influence sur la précision de la profondeur estimée.



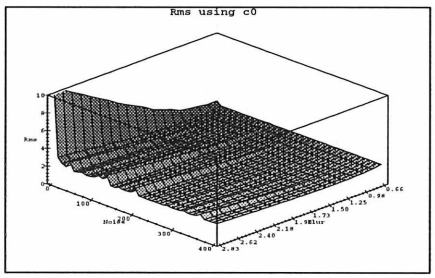
a. Signal lissé ($\sigma = 1$)



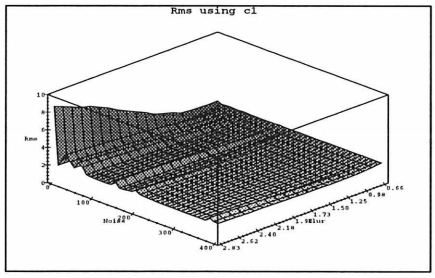
b. Signal lissé ($\sigma = 3$)



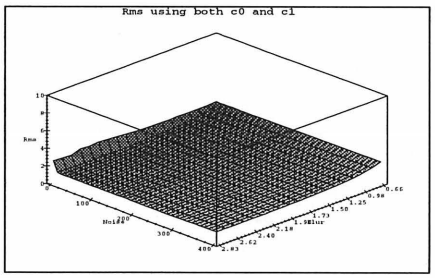
c. RMS en fonction du bruit



d. RMS pour c_0 en fonction du bruit et du flou

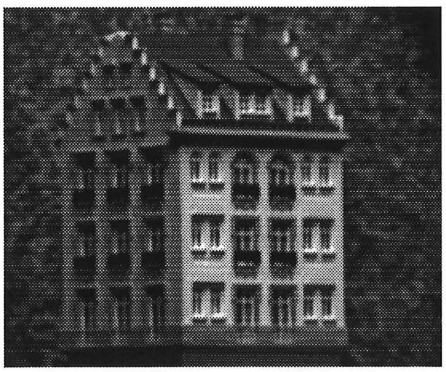


e. RMS pour c_1 en fonction du bruit et du flou

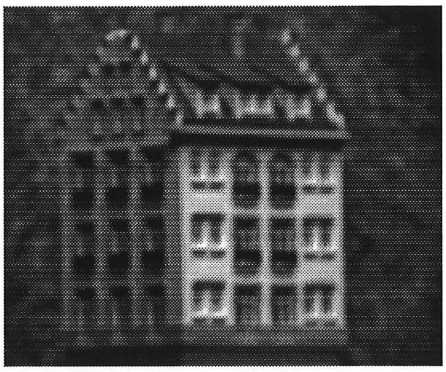


f. RMS modèle unifié en fonction du bruit et du flou

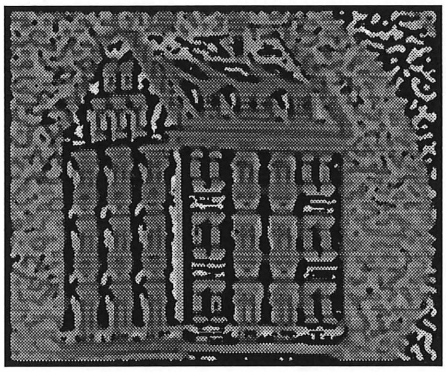
Figure 3.3 – *Estimation du flou à partir d'un signal 1D bruité*



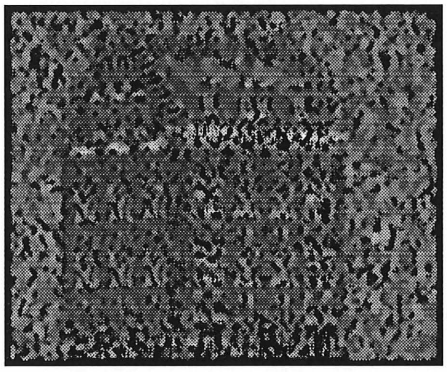
a. Image la moins floue



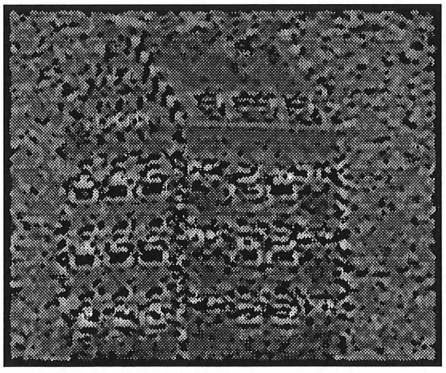
b. Image la plus floue



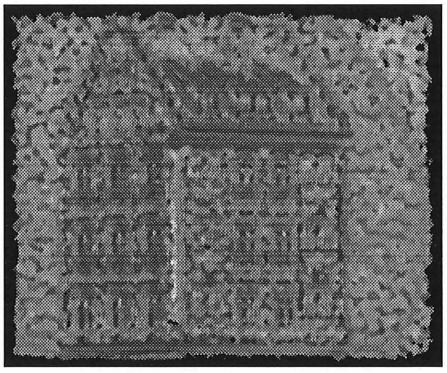
c. Flou avec $c_{0,0}$



d. Flou avec $c_{0,1}$

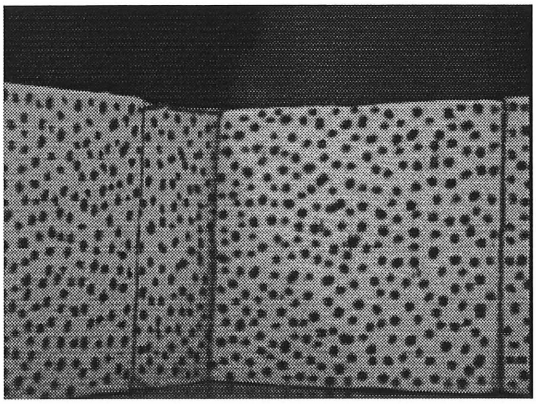


e. Flou avec $c_{1,0}$

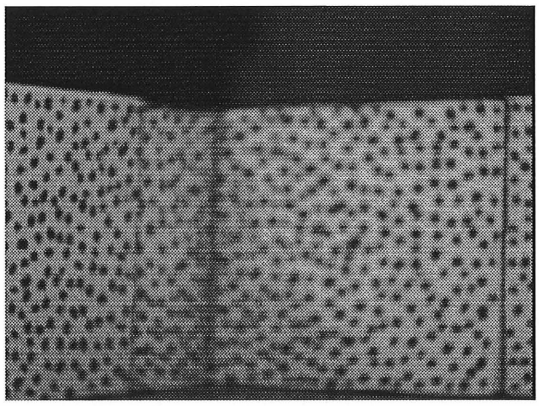


f. Flou avec modèle unifié

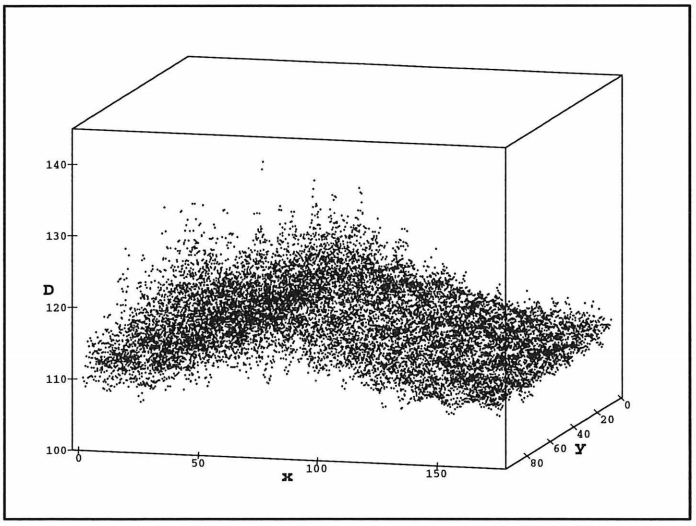
Figure 3.4 – *Estimation du flou à partir d'images réelles*



a. Image originale la moins floue

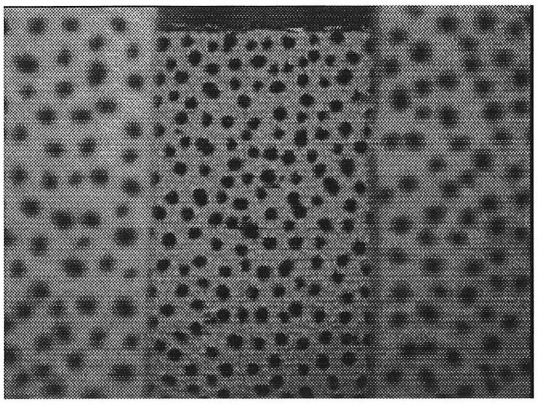


b. Image originale la plus floue

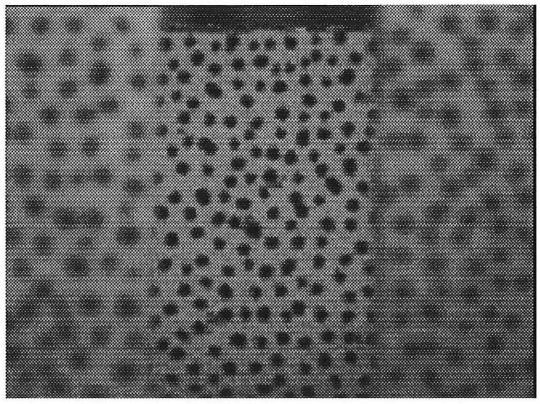


c. Profondeur estimée

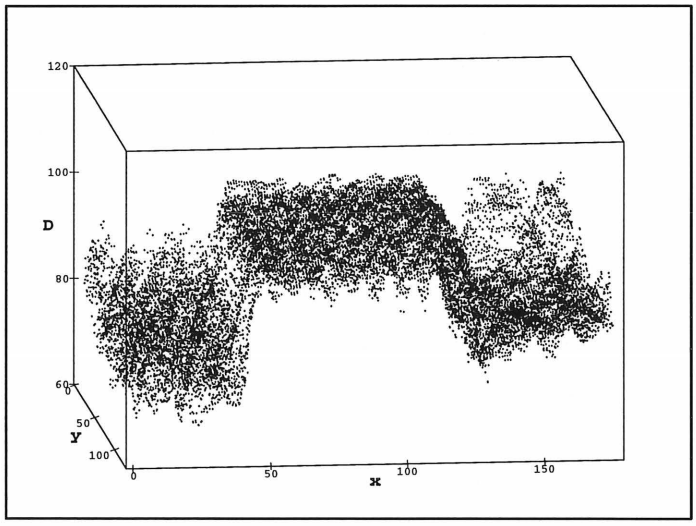
Figure 3.5 – *Profondeur d'une scène formée de deux plans obliques*



a. Image originale la moins floue

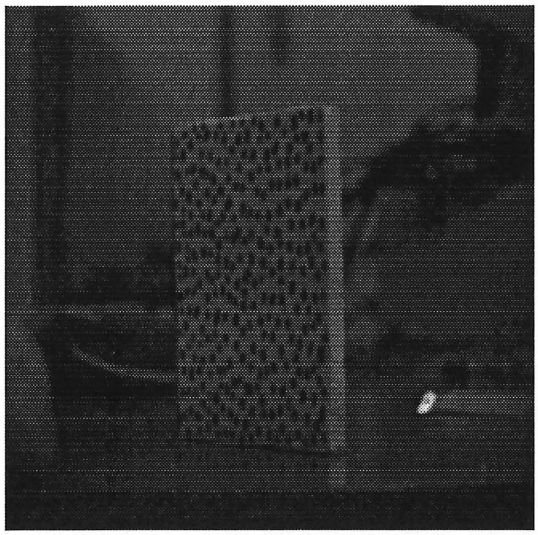


b. Image originale la plus floue

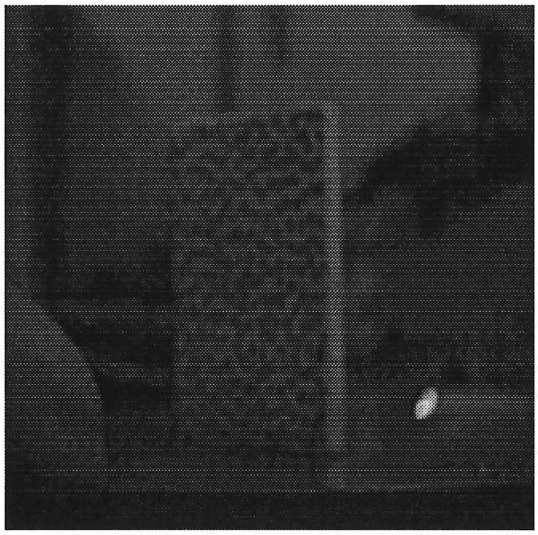


c. Profondeur estimée

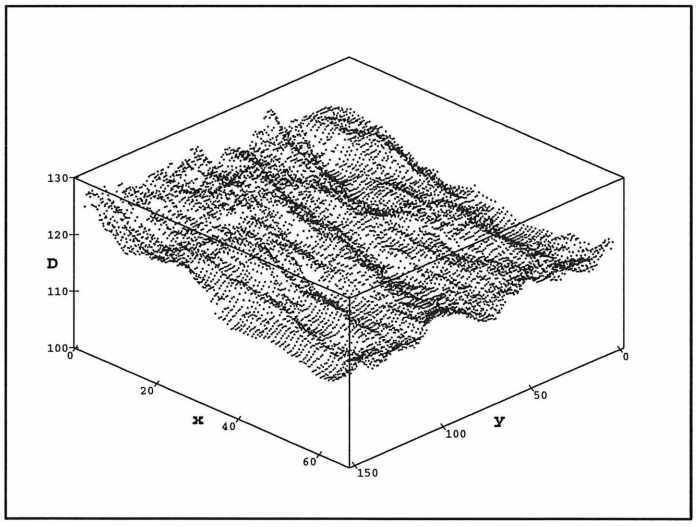
Figure 3.6 – *Profondeur d'une scène comprenant des discontinuités de flou*



a. Image originale la moins floue



b. Image originale la plus floue



c. Profondeur estimée

Figure 3.7 – *Profondeur d'une scène contenant un objet plan*

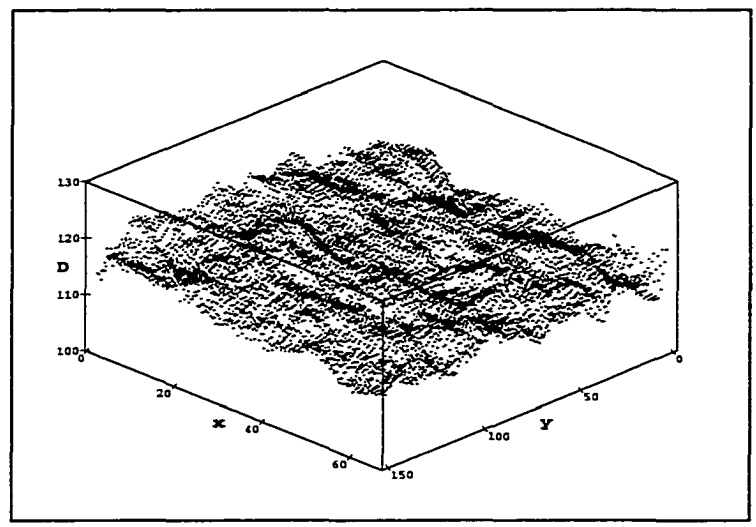


Figure 3.8 – *Profondeur estimée uniquement à l'aide de* $c_{0,0}$

CHAPITRE 4

MODÈLE UNIFIÉ POUR L'ESTIMATION DES INDICES DE PROFONDEUR

4.1 Introduction

L'un des objectifs de l'extraction des caractéristiques d'images dédiées à la perception 3D est la construction d'une représentation riche et compacte de la scène. Considérons une séquence d'images d'une scène réelle acquise par une caméra évoluant dans un espace 3D et dont les valeurs des paramètres extrinsèques (position et orientation) et intrinsèques (ouverture, distance focale, lentille, etc.) en fonction du temps sont connues. À partir de cette séquence d'images, il est possible d'extraire certains indices de profondeur, tels que la disparité, le flou et le mouvement 2D. La disparité est définie comme étant la distance entre deux points correspondants d'une paire d'images stéréoscopiques lorsque celles-ci sont superposées (figure 4.1). Les définitions du mouvement 2D et du flou ont

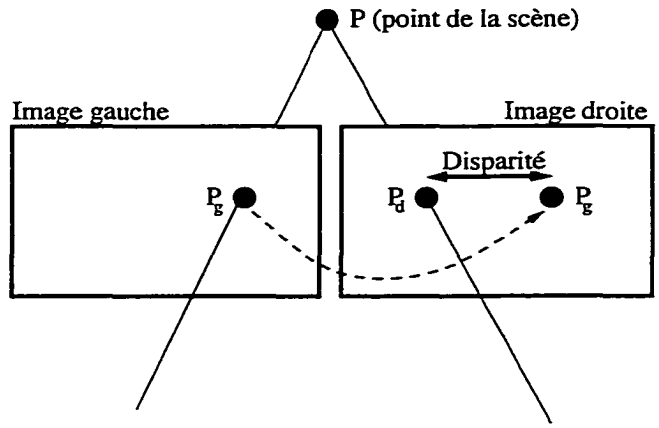


Figure 4.1 – *Disparité entre deux points correspondants*

été respectivement présentées aux chapitres 2 et 3.

Les techniques généralement proposées se contentent de calculer ces différents indices de profondeur indépendamment (se référer à [14, 34, 41], ainsi qu'aux sections 2.4 et 3.2). La complémentarité de ces informations n'est pas considérée, ce qui peut occasionner une perte de précision et de densité. Il serait intéressant d'estimer de manière simultanée et coopérative la disparité, le flou et le mouvement 2D. Aucune approche ne semble avoir été proposée en ce sens. En fait, il existe des méthodes permettant de calculer de manière synchrone le flot optique et le flou [30], ainsi que le flot optique et la disparité [52], mais aucune d'entre elles n'estime la disparité et le flou ou bien les trois indices simultanément.

Ce chapitre présente un algorithme permettant de calculer simultanément la différence de flou et la disparité entre une paire d'images (stéréoscopiques ou séquence) unidimensionnelles. Pour ce faire, nous généraliserons l'approche d'estimation du flou présentée dans le chapitre précédent. Sous les mêmes hypothèses (projection perspective, système de formation d'image passif, PSF gaussienne et $\sigma(x,y)$ localement constant), nous démontrerons que tout coefficient du polynôme d'Hermite, calculé à partir de l'image la plus floue, peut-être exprimé en fonction des dérivées partielles de la seconde image, d'une

différence de flou et d'un facteur de décalage. Il est ainsi possible d'estimer ces derniers en résolvant un système d'équations. Les estimations résultantes sont ensuite régularisées afin de raffiner la solution finale et d'améliorer ainsi la précision.

La section suivante détaille les règles d'estimation simultanée du flou et de la disparité pour les images 1D. L'algorithme résultant est décrit à la section 4.3. Les résultats expérimentaux et l'influence des différents paramètres sont ensuite présentés à la section 4.4.

4.2 Estimation simultanée du flou et de la disparité

4.2.1 Approche basée sur la transformée d'Hermite

Cette section présente le développement mathématique sur lequel repose l'approche d'estimation simultanée du flou et de la disparité dans le cas d'un signal 1D. Ce développement est en fait une généralisation de la méthode d'estimation du flou présentée au chapitre 3. Considérons $I_r(x_r)$ et $I_l(x_l)$, respectivement les images droite et gauche d'une même scène, obtenues en modifiant un ou plusieurs paramètres intrinsèques des caméras (figure 3.1) d'un système d'acquisition stéréoscopique. Supposons que $I_r(x_r)$ est plus floue que $I_l(x_l)$. Tel que mentionné au chapitre précédent, une telle supposition n'a aucune conséquence, puisqu'il est possible de déterminer quelle image est la plus floue. Lorsque la fonction PSF est une gaussienne, $I_r(x_r) = (I * g_{\sigma_r})(x_r)$ et $I_l(x_l) = (I * g_{\sigma_l})(x_l)$, où $g_\sigma(x)$ est une gaussienne de variance σ^2, $I(x)$ est l'image en focus, $*$ la convolution et $\sigma_r > \sigma_l$. Ainsi, la relation qui existe entre $I_r(x_r)$ et $I_l(x_l)$ est donnée par:

$$I_r(x_r) = (I_l * g_\beta)(x_l) \tag{4.1}$$

où $x_l = x_r + s$ représente la relation entre les deux systèmes de coordonnées, s la disparité et $\beta = \sqrt{\sigma_r^2 - \sigma_l^2}$ la différence de flou entre les deux images.

L'estimation du flou et de la disparité consiste donc à calculer β et s. À cette fin, utilisons la transformée d'Hermite présentée au chapitre précédent. L'approximation de l'image $I_r(x_r)$ par la base polynomiale $\{P_n(x_r)\}$ à l'intérieur d'une fenêtre $g_\sigma(x_r)$ est donnée par :

$$I_r(x_r) = \sum_{n=0}^{+\infty} r_n(m) P_n(x_r - m), \tag{4.2}$$

où les coefficients $r_n(m)$ sont définis par :

$$r_n(m) = (I_r * h_n)(m) \tag{4.3}$$

et

$$h_n(x) = P_n(x) g_\sigma(x) = (-1)^n \frac{d^n}{d(\frac{x}{\sigma})^n} \frac{1}{\sqrt{\pi}\sigma} e^{-\frac{x^2}{\sigma^2}}. \tag{4.4}$$

En remplaçant la fenêtre $g(x_r)$ par $w(x_r) = \sum_{k=-K}^{K} \sqrt{g_\sigma(x_r - kT)}$ et en considérant un espacement d'échantillonnage T équidistant, l'approximation de l'image $I_r(x_r)$ à l'intérieur de cette fenêtre centrée en x_r devient :

$$I_r(x_r) w(x_r) = \sum_{k=-K}^{K} I_r(x_r) \sqrt{g_\sigma(x_r - kT)}. \tag{4.5}$$

En substituant $I_r(x_r)$ de la partie de droite par l'équation (4.2), nous obtenons :

$$I_r(x_r) = \sum_{n=0}^{\infty} \sum_{k=-K}^{K} r_n(kT) \frac{P_n(x_r - kT)\sqrt{g_\sigma(x_r - kT)}}{w(x_r)}. \tag{4.6}$$

En considérant que $r_n(x_r)$ représente les coefficients de l'image droite et $l_n(x_l)$ ceux de l'image gauche, selon l'équation (4.3), la relation $r_n(x_r) = l_n(x_l)$ doit être satisfaite. En introduisant l'équation (4.1) dans l'équation (4.3), les coefficients de l'image $I_r(x_r)$ peuvent être exprimés sous la forme :

$$r_n(x_r) = I_l(x_l) * g_\beta(x_l) * h_n(x_r) = (I_l * f_n)(x_l), \tag{4.7}$$

où

$$f_n(x) = (g_\beta * h_n)(x) = (-1)^n \sigma^n \frac{d^n}{dx^n} \frac{1}{\sqrt{\pi}\alpha} e^{-\frac{x^2}{\alpha^2}} \tag{4.8}$$

et $\alpha^2 = \sigma^2 + \beta^2$. En d'autres termes, l'équation (4.7) peut s'écrire :

$$r_n(x_r) = \int_{-\infty}^{+\infty} I_l(x_r + s - x) f_n(x) dx. \tag{4.9}$$

Les équations (4.3) et (4.9) illustrent la relation qui existe entre les images I_r et I_l. Cependant, nous nous retrouvons avec une équation non linéaire à deux inconnues (β et s). Nous devons donc simplifier cette relation afin d'en extraire une règle pour le calcul simultané du flou β et de la disparité s. Considérons le développement en séries de Taylor de $I_l(x_r - x + s)$ au point x_r :

$$I_l(x_r + s - x) = \sum_{p=0}^{+\infty} \frac{(s-x)^p I_l^{(p)}(x_r)}{p!}, \tag{4.10}$$

où $I_l^{(p)}(x)$ représente la p^e dérivée de $I_l(x)$. En combinant les équations (4.9) et (4.10), nous obtenons :

$$r_n(x_r) = \sum_{p=0}^{+\infty} s_{n,p} I_l^{(p)}(x_r), \tag{4.11}$$

où

$$s_{n,p} = \int_{-\infty}^{+\infty} \frac{(s-x)^p}{p!} f_n(x) dx. \tag{4.12}$$

Les équations (4.3) et (4.11) expriment la relation qui prévaut entre les images I_r et I_l. Le coefficient $s_{n,p}$ de cette dernière est proportionnel au p^e moment de $f_n(x)$ centré sur s et est une fonction du flou β et de la disparité s. Cela signifie qu'il est possible d'estimer la relation entre β et s à partir de $I_r(x)$, $I_l(x)$ et $f_n(x)$. Afin de réduire les coûts reliés au calcul des moments, développons une procédure de calcul rapide pour $s_{n,p}$. Sachant que :

$$s_{n,p} = \int_{-\infty}^{+\infty} \frac{(s-x)^p}{p!} f_n(x) dx = \sum_{i=0}^{p} C_i^p s^i \int_{-\infty}^{+\infty} \frac{(-x)^{p-i}}{p!} f_n(x) dx = \frac{1}{p!} \sum_{i=0}^{p} C_i^p s^i a_{n,p-i}, \tag{4.13}$$

où C_r^n est la combinaison de r éléments parmi n et :

$$a_{n,p-i} = \int_{-\infty}^{+\infty} (-x)^{p-i} f_n(x) dx, \tag{4.14}$$

nous distinguons les cinq situations suivantes :

- Si $n = p = 0$, alors l'équation (4.12) donne $s_{0,0} = 1$.
- Si $p = 0$ et $n > 0$, alors $s_{n,0} = \int_{-\infty}^{+\infty} f_n(x) dx = 0$.
- Si $n = 0$ et $p = 2r + 1$, alors $a_{0,2r+1} = \int_{-\infty}^{+\infty} x^{2r+1} f_0(x) dx = 0$. D'où :

$$s_{0,2r+1} = \frac{1}{(2r+1)!} \sum_{i=0}^{r} C_{2i+1}^{2r+1} s^{2i+1} a_{0,2r-2i}. \tag{4.15}$$

- Si $n = 0$ et $p = 2r$, alors $a_{0,2r} = \int_{-\infty}^{+\infty} x^{2r} f_0(x) dx = \frac{(2r-1)(2r-3)\cdots 3\alpha^{2r}}{2^r}$ (voir Annexe B) et :

$$s_{0,2r} = \frac{1}{(2r)!} \sum_{i=0}^{r} C_{2i}^{2r} s^{2i} a_{0,2r-2i}. \tag{4.16}$$

- Si $n > 0$ et $p > 0$, à l'aide de la règle d'intégration par parties, l'équation (4.14) peut s'écrire :

$$a_{n,p-i} = -\sigma(p-i)(-1)^{p-i-1} \int_{-\infty}^{+\infty} x^{p-i-1} f_{n-1}(x) dx = -\sigma(p-i) a_{n-1,p-i-1}. \tag{4.17}$$

Cette équation fournit une méthode récursive de calcul de $a_{n,p-i}$ pouvant être simplifiée comme suit :

$$a_{n,p-i} = \begin{cases} (-1)^n \sigma^n \frac{(p-i)!}{(p-i-n)!} \frac{(p-i-n-1)(p-i-n-3)\cdots 3\alpha^{p-i-n}}{2^{(p-i-n)/2}} & \text{si } n \leq p-i \text{ et} \\ & p-i-n \text{ est pair} \\ 0 & \text{sinon.} \end{cases} \tag{4.18}$$

Ainsi :

$$s_{n,p} = \begin{cases} 0 & \text{si } n > p \\ \frac{1}{p!} \sum_{i=0}^{r} C_{2i}^{p} s^{2i} a_{n,p-2i} & \text{si } p-n = 2r \\ \frac{1}{p!} \sum_{i=0}^{r} C_{2i+1}^{p} s^{2i+1} a_{n,p-(2i+1)} & \text{si } p-n = 2r+1. \end{cases} \tag{4.19}$$

En conclusion, le calcul $s_{n,p}$ est directement donné par :

$$s_{n,p} = \begin{cases} (-1)^n \sigma^n \frac{1}{p!} \sum_{i=0}^{r} \frac{(p-2i)!}{(2r-2i)!} C_{2i}^{p} s^{2i} a_{0,2r-2i} & \text{si } p > n \text{ et } p-n = 2r \\ (-1)^n \sigma^n \frac{1}{p!} \sum_{i=0}^{r} \frac{(p-(2i+1))!}{(2r-2i)!} C_{2i+1}^{p} s^{2i+1} a_{0,2r-2i} & \text{si } p > n \text{ et } p-n = 2r+1 \\ (-1)^n \sigma^n & \text{si } n = p \\ 0 & \text{sinon.} \end{cases} \tag{4.20}$$

Cette équation constitue une règle simple et efficace pour déterminer la valeur de $s_{n,p}$. Elle permet ainsi la simplification des calculs de $r_n(x_r)$ et l'approximation de $I_r(x_r)$. En utilisant les équations (4.3), (4.11) et (4.20), la relation entre les deux images $I_r(x)$ et $I_l(x)$ est désormais donnée par :

$$r_n(x_r) = \sum_{p \geq n}^{+\infty} s_{n,p} I_l^{(p)}(x_r) = (I_r * h_n)(x_r). \tag{4.21}$$

L'équation de reconstruction de l'image $I_r(x)$ (éq. (4.6)) devient donc :

$$I_r(x) = \sum_{n=0}^{+\infty} \sum_{p \geq n}^{+\infty} \sum_{k=-K}^{K} s_{n,p} I_l^{(p)}(x) P_n(x - kT) \frac{\sqrt{g_\sigma(x - kT)}}{w(x)} \tag{4.22}$$

Il est maintenant possible de calculer la différence de flou et la disparité. À titre d'exemple, considérons l'utilisation des coefficients r_0 et r_1 dans l'équation (4.21). Dans le cas de r_0, nous supposons que $\sum_{p>3} s_{0,p} I_l^{(p)}(x_r) = 0$, tandis que pour r_1, $\sum_{p>3}^{+\infty} s_{1,p} I_l^{(p)}(x_r) = 0$. Selon l'équation (4.20), $s_{0,0} = 1$, $s_{0,1} = s$, $s_{0,2} = \frac{\alpha^2}{4} + \frac{s^2}{2}$, $s_{0,3} = s(\frac{\alpha^2}{4} + \frac{s^2}{6})$, $s_{1,1} = -\sigma$, $s_{1,2} = -s\sigma$ et $s_{1,3} = -\sigma(\frac{\alpha^2}{4} + \frac{s^2}{2})$. En se basant sur l'équation (4.21), nous obtenons :

$$\begin{cases} r_0(x_r) = I_l(x_r) + s I_l'(x_r) + (\frac{\alpha^2}{4} + \frac{s^2}{2}) I_l''(x_r) + s(\frac{\alpha^2}{4} + \frac{s^2}{6}) I_l'''(x_r) = (I_r * h_0)(x_r) \\ r_1(x_r) = -\sigma I_l'(x_r) - s\sigma I_l''(x_r) - \sigma(\frac{\alpha^2}{4} + \frac{s^2}{2}) I_l'''(x_r) = (I_r * h_1)(x_r), \end{cases} \tag{4.23}$$

où $h_0(x_r)$ et $h_1(x_r)$ correspondent respectivement à un filtre gaussien de variance $\sigma^2/2$ et à sa première dérivée multipliée par un facteur constant $(-\sigma)$ (équation (4.4)). Sachant que $\alpha^2 = \beta^2 + \sigma^2$, nous nous retrouvons donc avec deux équations à deux inconnues (β et s).

La résolution d'un tel système d'équations ne mène toutefois pas à une solution unique et précise pour de nombreux points de l'image. En fait, plusieurs causes, dont l'imprécision reliée à l'approximation des images et l'imprécision du calcul des dérivées au niveau numérique, empêchent le respect de la relation (4.21). Pour cette raison, il est préférable de rechercher une solution qui minimise à la fois les erreurs suivantes :

$$\begin{cases} e_0(x_r) = (I_l(x_r) + sI_l'(x_r) + (\frac{\alpha^2}{4} + \frac{s^2}{2})I_l''(x_r) + s(\frac{\alpha^2}{4} + \frac{s^2}{6})I_l'''(x_r) - (I_r * h_0)(x_r))^2 \\ e_1(x_r) = (-\sigma I_l'(x_r) - s\sigma I_l''(x_r) - \sigma(\frac{\alpha^2}{4} + \frac{s^2}{2})I_l'''(x_r) - (I_r * h_1)(x_r))^2 \\ \vdots \\ e_n(x_r) = (\sum_{p\geq n}^{+\infty} s_{n,p} I_l^{(p)}(x_r) - (I_r * h_n)(x_r))^2. \end{cases} \tag{4.24}$$

Cela peut être effectué en minimisant :

$$e_c(x_r) = \sum_{m=-M}^{M} \sum_{i=0}^{n} e_i(x_r + m), \tag{4.25}$$

où $M \geq 0$ est un paramètre qui détermine la taille du voisinage. Pour chaque point x_r, il suffit donc de rechercher les couples (β,s) qui correspondent à des minima locaux de e_c, c'est-à-dire tels que :

$$\frac{\delta e_c}{\delta \beta} = 0, \quad \frac{\delta e_c}{\delta s} = 0, \quad \frac{\delta^2 e_c}{\delta \beta^2} > 0 \quad \text{et} \quad \frac{\delta^2 e_c}{\delta \beta^2}\frac{\delta^2 e_c}{\delta s^2} - \left(\frac{\delta^2 e_c}{\delta \beta \delta s}\right)^2 > 0 \tag{4.26}$$

Parmi les couples ainsi obtenus pour le point x_r, celui qui minimise la valeur de e_c est retenu. Cette démarche permet donc de déterminer simultanément la variation de flou β et la disparité s.

4.2.2 Régularisation de la solution

Les expérimentations ont montré que les équations (4.25) et (4.26) ne garantissent pas l'obtention d'une solution lisse. Cela contredit l'hypothèse selon laquelle le flou et la disparité sont localement constants (section 4.1). Il importe donc de raffiner la solution

initiale en pénalisant les variations brusques dans les fonctions $\beta(x)$ et $s(x)$. Notons que nous utiliserons uniquement β et s dans ce qui suit afin de simplifier la notation. Ainsi, cette pénalisation (contrainte de lissage) peut se traduire par la minimisation de l'intégrale suivante :

$$e_s = \int \left((\beta')^2 + (s')^2 \right) dx. \tag{4.27}$$

Par ailleurs, les β et s estimés doivent également minimiser l'erreur suivante :

$$e_e = \int (I_r(x-s) - (I_l * g_\beta)(x))^2 \, dx. \tag{4.28}$$

Il s'agit donc de minimiser e_e avec e_s comme contrainte, c'est-à-dire $e_e + \lambda e_s$:

$$\int (I_r(x-s) - (I_l * g_\beta)(x))^2 + \lambda \left((\beta')^2 + (s')^2 \right) dx, \tag{4.29}$$

où $*$ est l'opérateur de convolution et λ un paramètre qui pondère l'erreur par rapport au lissage. La valeur de ce dernier est généralement élevée pour les régions homogènes de β et s et faible dans le cas contraire.

Pour un point x quelconque, les équations itératives permettant de résoudre ce type de système d'équations sont données par [43] :

$$s^k = s^{k-1} - \eta_s \frac{\delta e_t}{\delta s}, \qquad \text{et} \qquad \beta^k = \beta^{k-1} - \eta_\beta \frac{\delta e_t}{\delta \beta} \tag{4.30}$$

où $e_t(x) = (I_r(x-s) - (I_l * g_\beta)(x))^2 + \lambda((\beta')^2 + (s')^2)$, η_s et η_β pondèrent la variation par rapport à la valeur précédente et k est le nombre d'itérations. Déterminons maintenant les valeurs de $\frac{\delta e_t}{\delta s}$ et $\frac{\delta e_t}{\delta \beta}$. Pour ce faire, posons $I_r(x-s) \simeq I_r(x) - sI_r'(x)$, puis approchons β' et s' par $\beta - \beta_{av}$ et $s - s_{av}$, respectivement. β_{av} et s_{av} représentent les valeurs moyennes de β et s dans un voisinage donné. Ainsi, e_t devient :

$$e_t(x) = (I_r(x) - sI_r'(x) - (I_l * g_\beta)(x))^2 + \lambda((\beta - \beta_{av})^2 + (s - s_{av})^2). \tag{4.31}$$

En dérivant cette dernière équation par rapport à s et β, nous obtenons :

$$\frac{\delta e_t}{\delta s} = 2\left[\left(I_r - sI_r' - (I_l * g_\beta)\right)\left(-I_r'\right) + \lambda(s - s_{av})\right] \tag{4.32}$$

et

$$\frac{\delta e_t}{\delta \beta} = 2\left[\left(I_r - sI_r' - (I_l * g_\beta)\right)\left(I_l * \frac{\delta g_\beta}{\delta \beta}\right) + \lambda(\beta - \beta_{av})\right] \tag{4.33}$$

où :

$$\frac{\delta g_\beta}{\delta \beta} = \frac{(2x^2 - \beta^2)}{\beta^4\sqrt{\pi}} e^{-x^2/\beta^2}.$$

Par conséquent, en combinant les équations (4.30), (4.32) et (4.33), nous obtenons :

$$s^k = s^{k-1} - \eta_s \times 2\left[\left(I_r - sI_r' - (I_l * g_\beta)\right)\left(-I_r'\right) + \lambda(s - s_{av})\right], \tag{4.34}$$

et

$$\beta^k = \beta^{k-1} - \eta_\beta \times 2\left[\left(I_r - sI_r' - (I_l * g_\beta)\right)\left(I_l * \frac{\delta g_\beta}{\delta \beta}\right) + \lambda(\beta - \beta_{av})\right] \tag{4.35}$$

où $k = 1 \cdots K$, $\eta_s > 0$ et $\eta_\beta > 0$. Notons que les valeurs de η_β et η_s ne sont pas nécessairement constantes pour tous les points de l'image.

4.3 Algorithme résultant

L'algorithme pour l'estimation simultanée de la différence de flou β et de la disparité s peut être spécifié directement à partir du développement mathématique présenté dans les sections précédentes. Considérons une somme finie pour l'équation (4.21) (c'est-à-dire que nous négligeons les termes d'ordre supérieur) :

$$r_n(x_r) = (I_r * h_n)(x_r) \tag{4.36}$$

et

$$r_n(x_r) = \sum_{p \geq n}^{P} s_{n,p} I_l^{(p)}(x_r), \tag{4.37}$$

où $n = 0,1,\cdots,N$. Pour un n donné, les équations (4.36) et (4.37) forment un système de deux équations à deux inconnues : r_n et $s_{n,p}$. $s_{n,p}$ est une fonction du flou β et de la disparité s, lesquels peuvent ainsi être estimés en résolvant un système d'équations formé par l'utilisation d'au moins deux valeurs de n. Cependant, tel que mentionné à la section 4.2.1, les imprécisions dues à l'approximation des images et au calcul des dérivées numériques empêchent le respect des relations (4.36) et (4.37). Nous estimons donc β et s en minimisant l'équation (4.25). Pour chaque pixel (x,y), nous obtenons zéro, un ou plusieurs couples (β,s) correspondant aux minima locaux de e_c. À partir de ces couples, nous retenons celui qui produit la plus petite erreur. Ce dernier constitue l'estimation initiale (β^0,s^0). Lorsque β^0 et s^0 ont été estimés pour tous les points, nous régularisons de manière itérative la solution afin de la rendre lisse (équations (4.34) et (4.35)).

En résumé, soient σ, T, N, P, M, λ, K, η_β et η_s, l'algorithme est composé de trois étapes :

1. Estimation des r_n (équation (4.36)).
2. Estimation des valeurs initiales (β^0,s^0) en résolvant le système d'équations (4.26).
3. Régularisation de la solution (équations (4.34) et (4.35)).

4.4 Évaluation des performances

4.4.1 Comportement de l'algorithme

L'algorithme proposé provient d'une généralisation de l'approche d'estimation du flou présentée au chapitre précédent. Pour cette raison, nous nous attendons à ce qu'il se

comporte de manière similaire (section 3.6.1). À cette fin, examinons le comportement du système d'équations (4.36) et (4.37) en présence d'images constantes, de contours de type marche et de lignes.

- Images constantes : les dérivées de l'image sont nulles et le système d'équations est réduit à $r_n(x_r) = s_{n,0} I_l(x_r) = (I_r * h_n)(x_r)$. Selon l'équation (4.20), $s_{n,0}$ vaut 1 si $n = 0$ et 0 dans le cas contraire. Ainsi, $r_0(x_r) = I_l(x_r) = (I_r * h_0)(x_r)$. Cette équation ne permet donc pas de calculer le flou et la disparité.
- Contours de type marche : pour une marche, $I_l^{(p)}(x_r) = 0$, si p est pair. Dans un tel cas, le système d'équations devient $r_n(x_r) = s_{n,0} I_l(x_r) + \sum_{2p+1 \geq n}^{P} s_{n,2p+1} I_l^{(2p+1)}(x_r) = (I_r * h_n)(x_r)$. L'estimation simultanée du flou et de la disparité est donc possible.
- Lignes : pour une ligne, $I_l^{(p)}(x_r) = 0$, si p est impair. Dans cette situation, le système d'équations devient $r_n(x_r) = \sum_{2p \geq n}^{P} s_{n,2p} I_l^{(2p)}(x_r) = (I_r * h_n)(x_r)$. Le calcul du flou et de la disparité est donc possible.

En conclusion, tout comme dans le cas du flou, une estimation locale au sein des régions constantes de l'image n'est pas suffisante pour le calcul de la disparité. Ceci est dû au fait que la convolution d'une image constante avec une distribution gaussienne normalisée (PSF) redonne l'image elle-même.

4.4.2 Résultats expérimentaux

L'algorithme proposé a été testé à l'aide d'images unidimensionnelles. Pour ces tests, les dérivées des signaux sont calculées à l'aide de la convolution de ces derniers avec les dérivées appropriées de la fonction gaussienne. Les valeurs de l'échelle σ et de l'espacement T sont toutes deux égales à 1. La figure 4.2 présente un exemple d'estimation simultanée du flou et de la disparité dans le cas d'un signal 1D synthétique perturbé par un bruit

blanc additif de moyenne zéro. Le signal de la figure 4.2b est plus flou ($\sigma = \sqrt{5}$) que celui de la figure 4.2a ($\sigma = 1$). De plus, il a été décalé d'un pixel vers la droite ($s = -1$). Les figures 4.2c et 4.2d présentent respectivement les estimations initiales du flou (β^0) et de la disparité (s^0) découlant de l'équation (4.25) avec comme paramètres $N = 1$, $P = 3$ et $M = 2$. Elles illustrent clairement l'aspect irrégulier de la solution initiale. Comme le montrent les figures 4.2e à 4.2h, les estimations ont été régularisées à l'aide des équations (4.34) et (4.35). Les paramètres employés sont $\lambda = 4$, $\eta_\beta = 0.04$, $\eta_s = 0.04$ et K variant de 10 à 200. Remarquons que la solution tend vers les valeurs réelles de flou et de disparité ($\beta = 2$ et $s = -1$). Notons également que l'absence de valeur au niveau des bords est due à l'utilisation des masques de convolution lors du calcul des dérivées. La figure 4.3 montre les résultats d'un second test effectué à l'aide du signal de la figure 4.2a. Dans ce cas-ci, le deuxième signal de la paire est décalé de 1 pixel vers la droite ($s = -1$) et il comporte une discontinuité de flou de type marche ($\beta = 1$ et $\beta = 2$). Les estimations résultantes sont présentées dans les figures 4.3a à 4.3h. Les paramètres utilisés sont respectivement $N = 1$, $P = 3$, $M = 2$, $\lambda = 4$, $K \in \{10{,}50{,}200\}$, $\eta_\beta = 0.04$ et $\eta_s = 0.04$. D'après ces graphiques, nous constatons que la discontinuité de profondeur de type marche au niveau du flou devient évidente suite au processus de régularisation. Nous remarquons également que la solution finale est précise. En d'autres termes, elle s'approche des valeurs réelles de flou et de disparité.

Les performances de cet algorithme ont aussi été évaluées à l'aide d'images 1D créées à partir d'un signal réel. Les figures 4.4a et 4.4b présentent respectivement un signal réel et une version décalée ($s = -1$), puis lissée ($\beta = 1$) de ce dernier. Les graphiques représentant les estimations initiales du flou (β^0) et de la disparité (s^0) sont illustrés dans les figures 4.4c et 4.4d. Les paramètres de l'équation (4.25) que nous avons utilisés sont $N = 1$, $P = 3$ et $M = 2$. Encore une fois, notons l'aspect irrégulier de cette solution. Pour leur part, les figures 4.4e à 4.4h montrent les résultats du processus de régularisation

($\lambda = 4$, $K \in \{10,200\}$, $\eta_\beta = 0.01$ et $\eta_s = 0.01$). De manière générale, il semble que les valeurs de β et s tendent respectivement vers 1 et -1 au fur et à mesure que le nombre d'itérations augmente. Cela est conforme aux attentes puisque ce sont les valeurs réelles de flou et de disparité. Nous remarquons toutefois la présence de valeurs aberrantes pour ces deux indices principalement dans le voisinage du point $x = 70$. En analysant l'image originale (figure 4.4a), nous constatons que ce point correspond à une variation brusque et grande du niveau de gris. Étant donné que le processus de régularisation repose sur l'emploi de dérivées numériques, l'imprécision de ces dernières peut être l'une des causes de ce problème. En ce sens, une étude plus approfondie reste à effectuer. Finalement, la figure 4.5 montre les résultats fournis par notre algorithme lorsque l'image de la figure 4.4a est modifiée de sorte que $s = -2$ et $\beta = 2$. Les paramètres utilisés sont $N = 1$, $P = 3$, $M = 2$, $\lambda = 4$, $K \in \{100,200,500\}$, $\eta_\beta = 0.01$ et $\eta_s = 0.01$. La profondeur et le flou estimés pour chacun des pixels x sont tracés dans les figures 4.5a à 4.5h. Globalement, β^k et s^k tendent respectivement vers 2 et -2, ce qui signifie qu'ils se rapprochent graduellement des valeurs réelles de flou et de disparité. Cependant, nous constatons également la présence de valeurs aberrantes.

Examinons maintenant l'influence de la valeur des indices de profondeur sur leur erreur d'estimation respective. À cet effet, le tableau 4.1 présente, pour différentes valeurs de flou, les proportions de pixels dont l'erreur d'estimation de β se situe en deçà d'un certain seuil. Ces résultats ont été obtenus à partir du signal de la figure 4.2a en utilisant différentes valeurs β et s. Puisque seuls les points appartenant aux bords de l'image n'ont pas été considérés, les proportions affichées indiquent également la densité de la solution. Dans tous les cas, les paramètres utilisés sont $N = 1$, $P = 3$, $M = 2$, $\lambda = 4$, $K = 500$, $\eta_\beta = 0.04$ et $\eta_s = 0.04$. En analysant ces résultats, nous constatons que l'erreur d'estimation du flou demeure faible lorsque la disparité est de 1 pixel. Cependant, l'accroissement de cette dernière occasionne une hausse drastique du taux d'erreur, tel

	$\leq 1\%$	$\leq 2\%$	$\leq 5\%$	$\leq 10\%$	$\leq 50\%$
$\beta = 1, s = -1$	100%	100%	100%	100%	100%
$\beta = 2, s = -1$	100%	100%	100%	100%	100%
$\beta = 5, s = -1$	99%	100%	100%	100%	100%
$\beta = 10, s = -1$	96%	100%	100%	100%	100%
$\beta = 1, s = -2$	0%	0%	0%	0%	38%
$\beta = 1, s = -3$	0%	0%	0%	0%	0%

Tableau 4.1 – *Proportions des pixels dont l'erreur pour β est inférieure à un seuil donné*

qu'en témoigne le passage de $s = -1$ à $s = -2$. Pour sa part, le tableau 4.2 présente, pour différentes valeurs de disparité, les proportions de pixels dont l'erreur d'estimation de s (en pixels) est inférieure à un seuil donné. Notons que le signal et les paramètres sont

	0 pixel	≤ 1 pixel	≤ 2 pixels	≤ 3 pixels
$\beta = 1, s = -1$	100%	100%	100%	100%
$\beta = 2, s = -1$	100%	100%	100%	100%
$\beta = 5, s = -1$	100%	100%	100%	100%
$\beta = 10, s = -1$	100%	100%	100%	100%
$\beta = 1, s = -2$	100%	100%	100%	100%
$\beta = 1, s = -3$	0%	43,8%	95%	100%

Tableau 4.2 – *Proportions des pixels dont l'erreur pour s est inférieure à un seuil donné*

les mêmes que ceux utilisés pour les résultats du tableau précédent. Nous remarquons que la valeur de β ne semble pas avoir d'influence directe sur l'erreur d'estimation de s. En contrepartie, cette erreur augmente rapidement lorsque la disparité dépasse -2. En combinant les résultats des tableaux 4.1 et 4.2, nous pouvons donc conclure que l'algorithme estime de manière précise le flou et la disparité à condition que cette dernière soit inférieure ou égale à un pixel. Mentionnons toutefois qu'une telle limitation peut généralement être contournée grâce à l'utilisation du multi-résolution [1].

4.4.3 Influence des paramètres

Les choix des paramètres T, N, P, M, K, λ, η_β et η_s influencent la qualité des résultats. Étant donné que T, N et P découlent de l'approche développée pour l'estimation du flou, les valeurs devant être utilisées ont été discutées dans la section 3.6.3. Pour leur part, λ et K contrôlent le processus de raffinement itératif. Puisqu'un tel processus est également utilisé par de nombreux algorithmes d'estimation du flot optique, la description de leur influence respective est présentée dans la section 2.7.2. Dans ce qui suit, nous nous concentrerons donc sur les paramètres M, η_β et η_s.

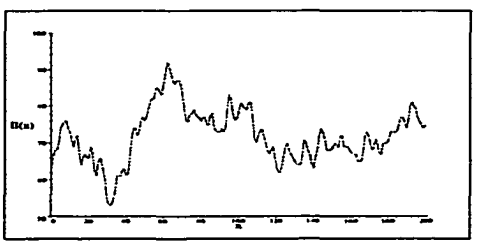
Le paramètre M détermine la taille du voisinage considéré lors de la minimisation de l'équation (4.25). Quand M est grand, les indices estimés pour un point donné sont proches de ceux de ses voisins. Cependant, puisque les indices de profondeur sont considérés comme étant localement constants, le voisinage utilisé doit être petit. Le choix de M constitue donc un compromis entre le degré de lissage et la précision des valeurs β et s estimées.

Selon les équations (4.34) et (4.35), les paramètres η_β et η_s pondèrent respectivement les variations de β et de s. Plus les valeurs de ces paramètres sont grandes, plus les variations sont amplifiées. Sachant que de petites variations peuvent occasionner une convergence très lente de la solution, il peut sembler important d'utiliser de grands η_β et η_s. Toutefois, cela peut faire osciller les estimations de β et de s autour du minimum recherché sans jamais converger. Par conséquent, le choix de η_β et η_s doit assurer un compromis entre la rapidité de convergence et la précision des valeurs β et s associées au minimum. Il existe certaines règles de calcul de η_β et de η_s afin d'assurer ce compromis [43].

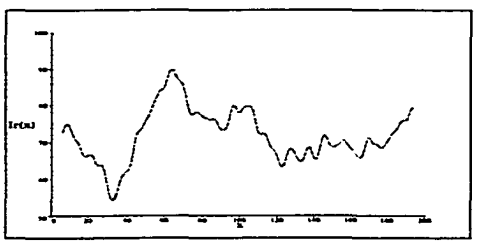
4.5 Conclusion

Les indices de profondeur contenus dans une ou plusieurs images sont fréquemment utilisés pour estimer la structure tridimensionnelle d'une scène réelle. À cet effet, nous avons proposé un modèle unifié permettant de calculer de manière simultanée et coopérative le flou et la disparité à partir d'images stéréoscopiques unidimensionnelles. En considérant une paire d'images d'une même scène provenant d'un système d'acquisition stéréoscopique dont les paramètres intrinsèques des caméras sont différents, nous avons généralisé l'approche d'estimation du flou présentée au chapitre précédent. En ce sens, nous avons démontré que tout coefficient du polynôme d'Hermite, calculé à partir de l'image la plus floue de la paire stéréoscopique, peut-être exprimé en fonction des dérivées partielles de la seconde image, d'une variation de flou et d'un facteur de décalage. Un système d'équations peut ainsi être utilisé afin d'estimer ces derniers. L'approche résultante se divise en trois étapes. Dans un premier temps, des approximations des images sont créées à l'aide d'une base du polynôme d'Hermite. Dans un deuxième temps, la résolution du système d'équations, ou plutôt la minimisation de son erreur, permet d'obtenir des valeurs initiales β^0 et s^0 pour chaque pixel (x,y). Finalement, cette solution est régularisée.

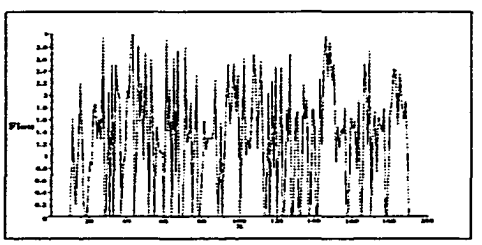
Les résultats fournis par notre algorithme confirment qu'il est possible d'estimer simultanément différents indices de profondeur : le flou et la disparité dans ce cas-ci. Ces indices peuvent servir, entre autres, à reconstruire de manière précise une scène 3D. Des travaux ultérieurs consisteront à étendre l'algorithme en 2D, à le valider, à gérer les grandes disparités, à inclure le mouvement, puis à fusionner les indices en vue de produire une carte de profondeurs unique.



a. Signal lissé ($\sigma = 1$)

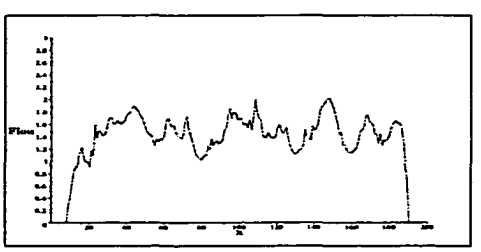


b. Signal modifié ($\sigma = \sqrt{5}$ et $s = -1$)

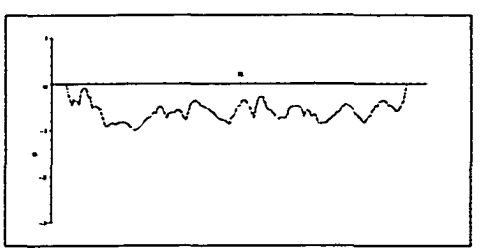


c. Flou initial (β^0)

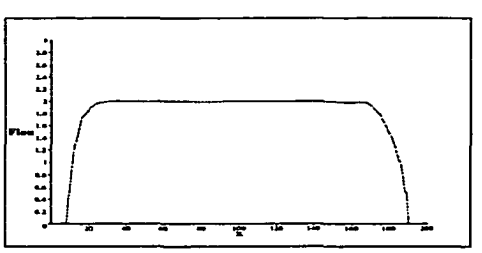
d. Disparité initiale (s^0)



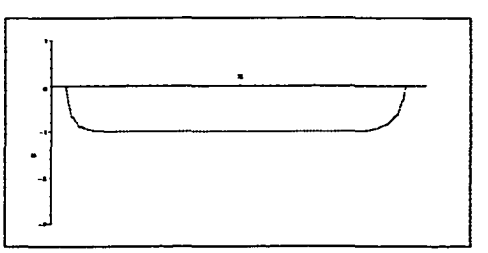
e. β^{10} (10^e itération)



f. s^{10} (10^e itération)

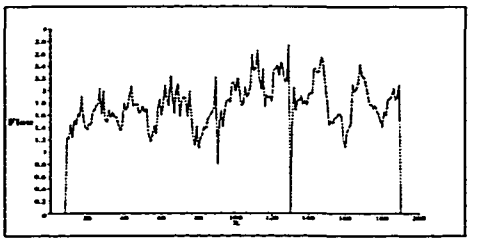


g. β^{200} (200^e itération)

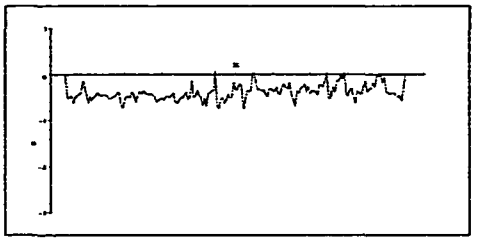


h. s^{200} (200^e itération)

Figure 4.2 – *Estimation du flou et de la disparité à partir d'un signal 1D bruité*

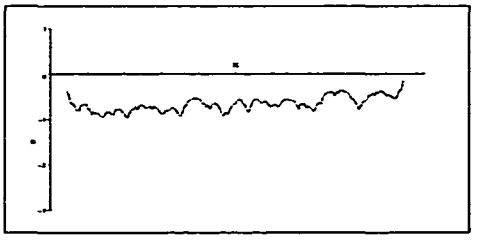


a. Flou initial (β^0)

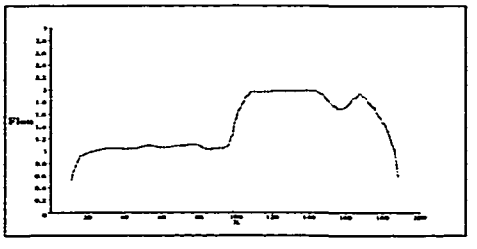


b. Disparité initiale (s^0)

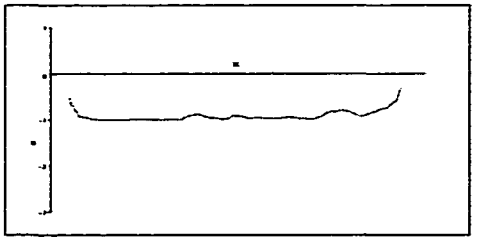
c. β^{10} (10^e itération)



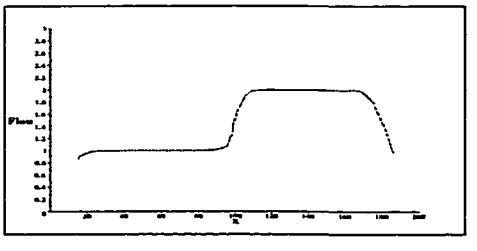
d. s^{10} (10^e itération)



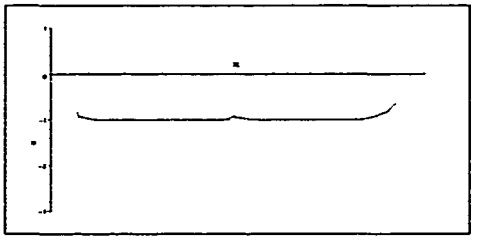
e. β^{50} (50^e itération)



f. s^{50} (50^e itération)

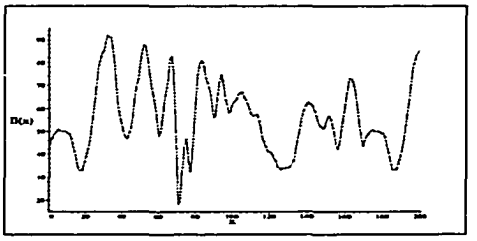


g. β^{200} (200^e itération)

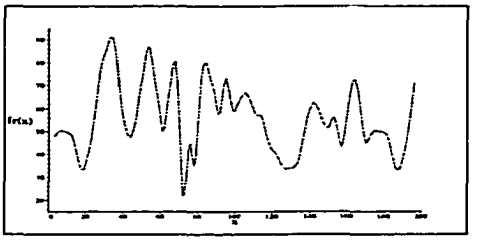


h. s^{200} (200^e itération)

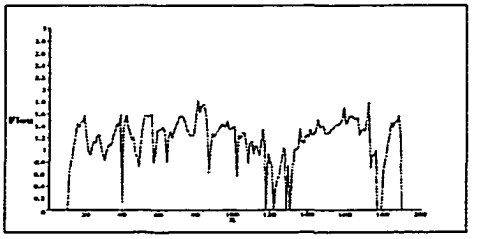
Figure 4.3 – *Flou et disparité d'un signal 1D contenant une discontinuité de flou*



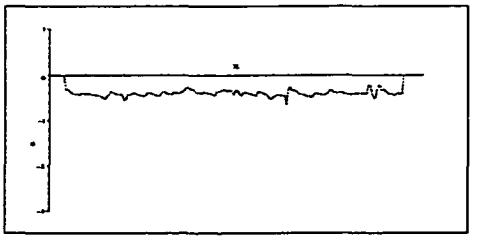
a. Image originale



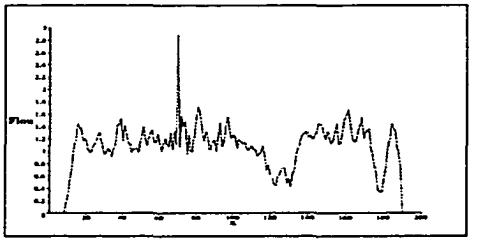
b. Image modifiée ($\beta = 1$ et $s = -1$)



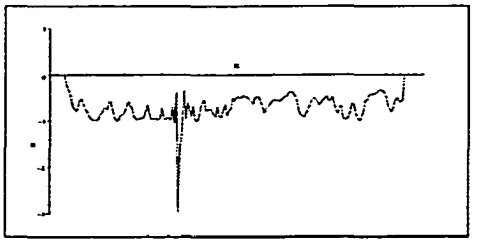
c. Flou initial (β^0)



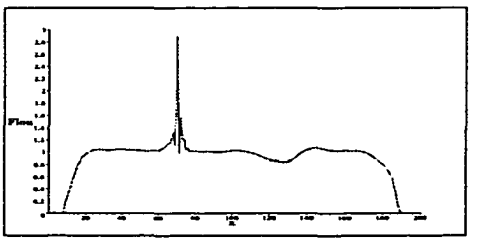
d. Disparité initiale (s^0)



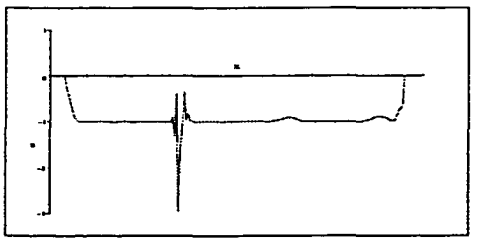
e. β^{10} (10^e itération)



f. s^{10} (10^e itération)

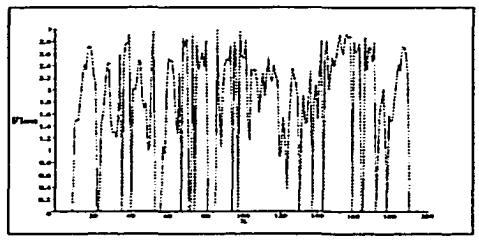


g. β^{200} (200^e itération)

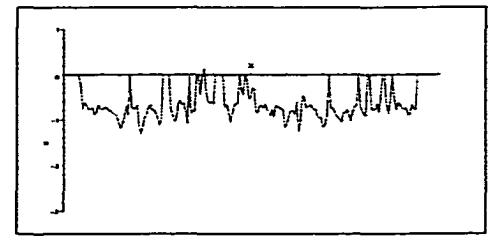


h. s^{200} (200^e itération)

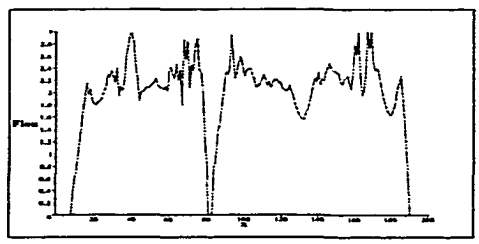
Figure 4.4 – *Flou et disparité à partir d'une image réelle 1D modifiée*



a. Flou initial (β^0)

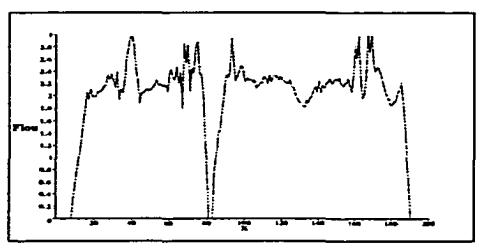


b. Disparité initiale (s^0)

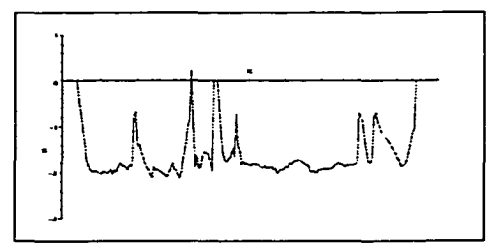


c. β^{100} (100^e itération)

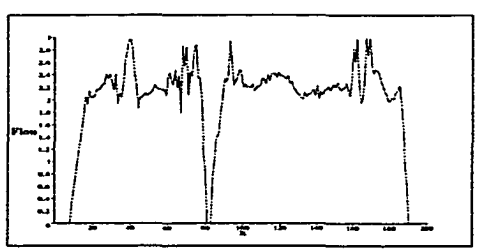
d. s^{100} (100^e itération)



e. β^{200} (200^e itération)



f. s^{200} (200^e itération)



g. β^{500} (500^e itération)

h. s^{500} (500^e itération)

Figure 4.5 – *Flou et disparité pour une image réelle 1D modifiée (test II)*

CONCLUSION

Dans le domaine de la reconstruction tridimensionnelle, différents indices associés à la perception 3D peuvent être extraits à partir d'une ou plusieurs images. À cet effet, nous avons proposé quatre approches différentes.

La première est une approche de détection des jonctions L, X, Y et T ainsi que des terminaisons de lignes. L'approche résultante est divisée en deux étapes. D'abord, à partir des informations de lignes extraites de l'image, la courbure locale est calculée. Par la suite, un traitement basé sur la courbure estimée et les informations du voisinage est effectué en vue de localiser avec précision les jonctions et extrémités de lignes. Les tests effectués ont montré que cet algorithme est efficace et qu'il permet de détecter les jonctions et terminaisons de lignes évidentes. Ils ont également montré que la localisation peut être accrue avec l'utilisation d'une mesure de courbure basée sur le taux de changement de direction des vecteurs d'orientation le long de la ligne. Cet algorithme a été validé dans une application de mise à jour automatique des bases de données cartographiques.

La seconde approche que nous avons présentée permet l'estimation du mouvement apparent en vue de déterminer l'ordre de profondeur relatif des objets de la scène. Deux étapes sont nécessaires : estimation du flot optique et segmentation des images selon un critère d'homogénéité appliqué au module des vecteurs de mouvement. Une estimation robuste du flot optique est donc primordiale. En ce sens, une évaluation qualitative des

algorithmes d'estimation du mouvement dans un contexte de segmentation d'images a préalablement été effectuée. Celle-ci a révélé qu'aucun des algorithmes testés ne réussit à résoudre entièrement les problèmes d'ouverture et d'occlusions. Cette évaluation nous a toutefois permis de choisir le meilleur algorithme (au sens de la simplicité, de la rapidité et de la qualité des résultats) et de l'améliorer. Les résultats fournis par notre approche ont confirmé que le flot optique peut être directement utilisé dans un processus de segmentation en couches des images.

La troisième approche que nous avons présentée permet de calculer le flou contenu dans une ou plusieurs images d'une même scène en vue d'estimer la profondeur des objets. Plus précisément, celle-ci calcule la différence de flou entre deux images acquises à partir d'une même caméra dont les paramètres intrinsèques ont été modifiés. Pour ce faire, les images sont approchées à l'aide d'une base du polynôme d'Hermite. Nous avons démontré que tout coefficient d'un tel polynôme, calculé à partir de l'image la plus floue, peut être exprimé en fonction des dérivées partielles de la seconde image et de la différence de flou. La résolution des systèmes d'équations permet donc d'obtenir différentes valeurs de flou, parmi lesquelles celle minimisant l'erreur quadratique est retenue. Les tests effectués ont clairement montré qu'une telle combinaison de différentes valeurs de flou mène à l'obtention d'une estimation dense et précise de ce dernier. Cela a d'ailleurs été validé par l'implantation d'une méthode d'estimation de la profondeur des points de la scène à partir du flou.

La dernière approche que nous avons présentée permet de calculer de manière simultanée et coopérative la disparité et le flou à partir de deux images unidimensionnelles (stéréoscopiques ou séquence). Elle constitue en fait une généralisation de notre approche d'estimation du flou. En ce sens, nous avons démontré qu'une image donnée de la paire peut être exprimée en fonction des dérivées partielles de la seconde image, d'une variation de flou et d'un facteur de décalage. Un système d'équations peut ainsi être utilisé

afin d'estimer ces derniers. La solution obtenue est ensuite régularisée afin de la rendre plus lisse et plus précise. Les résultats fournis par notre algorithme ont confirmé la possibilité d'estimer simultanément plusieurs indices de profondeur de manière précise. Par conséquent, la fusion de ces informations peut ensuite servir à assurer l'obtention d'une estimation précise de la profondeur de la scène.

En conclusion, toutes ces approches permettent d'obtenir directement ou indirectement des informations sur la structure tridimensionnelle d'une scène réelle. Elles pourront donc être insérées dans diverses applications appartenant à des domaines variés, tels que la médecine, le cinéma et la robotique. Des travaux ultérieurs concerneront l'unification des quatre indices de profondeur présentés dans ce document afin d'enrichir et de compléter le modèle de perception 3D.

ANNEXE A

Nous désirons montrer que la séquence $\{P_n(x) = H_n(\frac{x}{\sigma})\}$ est orthogonale pour $g_\sigma(x)$ (éq. (3.6)) définie sur l'intervalle $]-\infty, +\infty[$. Nous savons que la séquence $\{H_n(x)\}$ est orthogonale pour la fonction e^{-x^2} définie sur $]-\infty,\infty[$ [46] :

$$\int_{-\infty}^{+\infty} e^{-x^2} H_n(x) H_m(x) dx = 2^n n! \sqrt{\pi} \delta_{m,n}, \tag{4.38}$$

où $\delta_{m,n}$ est le delta de Kronecker. Considérons l'intégrale :

$$\int_{-\infty}^{+\infty} \frac{e^{-\frac{x^2}{\sigma^2}}}{\sqrt{\pi}\sigma} H_n(\frac{x}{\sigma}) H_m(\frac{x}{\sigma}) dx. \tag{4.39}$$

En effectuant le changement de variables $y = x/\sigma$, cette intégrale devient :

$$\frac{1}{\sqrt{\pi}} \int_{-\infty}^{+\infty} e^{-y^2} H_n(y) H_m(y) dy = 2^n n! \delta_{m,n}. \tag{4.40}$$

ANNEXE B

Nous désirons calculer le moment d'ordre n de la gaussienne $g_\sigma(x)$ (éq. (3.6)) :

$$\int_{-\infty}^{+\infty} x^n g_\sigma(x) dx. \tag{4.41}$$

Sachant que $x^{2r+1} g_\sigma(x)$ est impair, nous pouvons déduire que les moments impairs sont nuls. Lorsque $n = 2r$, en utilisant l'intégration par parties $u = g_\sigma(x)$ et $dv = x^{2r}$, nous obtenons :

$$\int_{-\infty}^{+\infty} x^{2r} g_\sigma(x) dx = \frac{2}{(2r+1)\sigma^2} \int_{-\infty}^{+\infty} x^{2r+2} g_\sigma(x) dx \tag{4.42}$$

$$\mu_{2r+2} = \frac{(2r+1)\sigma^2}{2} \mu_{2r} \tag{4.43}$$

$$\mu_{2r} = \frac{(2r-1)(2r-3)\cdots 3\sigma^{2r}}{2^r}. \tag{4.44}$$

ANNEXE C

Nous désirons déduire une règle de calcul de la profondeur à partir d'une différence de flou β^2 entre les deux images $I_c(x,y)$ et $I_b(x,y)$. Considérons l'image $I_c(x,y)$ acquise à l'aide d'une caméra optique dont les valeurs des paramètres sont v, F, f, et σ. Conformément à la géométrie d'une caméra (figure 3.1), la relation entre la profondeur z et le flou σ est donnée par :

$$k\sigma = dv(\frac{1}{F} - \frac{1}{z} - \frac{1}{v}), \tag{4.45}$$

où d représente le rayon de la lentille. Supposons que l'image $I_b(x,y)$ fut acquise en incrémentant v de δv, où $\delta v > 0$. Cela occasionne un changement de valeur pour σ, c'est-à-dire $\sigma + \delta\sigma$, où :

$$k\delta\sigma = d\delta v(\frac{1}{F} - \frac{1}{z}). \tag{4.46}$$

Selon l'équation (3.4), la relation entre β et σ est donnée par :

$$\beta^2 = 2\sigma\delta\sigma + (\delta\sigma)^2. \tag{4.47}$$

En combinant les équations (4.45), (4.46) et (4.47), nous obtenons :

$$d^2\delta v(2v + \delta v)x^2 - 2d^2\delta vx - k^2\beta^2 = 0, \tag{4.48}$$

où $x = \frac{1}{F} - \frac{1}{z}$. En résolvant cette équation du second ordre selon x, nous obtenons deux solutions, dont une seule est valide :

$$\frac{1}{z} = \frac{1}{F} - \frac{1}{2v + \delta v}(1 + \sqrt{1 + \frac{f^2(2v + \delta v)}{F^2 \delta v} k^2 \beta^2}). \tag{4.49}$$

Bibliographie

[1] P. Anandan. A Computational Framework and an Algorithm for the Measurement of Visual Motion. *International Journal of Computer Vision*, 2:283–310, 1989.

[2] D.H. Ballard and C.M. Brown. *Computer Vision.* Prentice Hall, Englewook Cliffs, New Jersey, US, 1982.

[3] J.L. Barron, D.J. Fleet, and S.S. Beauchemin. Performance of Optical Flow Techniques. *International Journal of Computer Vision*, 12:43–77, 1994.

[4] P. R. Beaudet. Rotationally invariant image operators. In *Proceedings of the International Joint Conference on Pattern Recognition*, pages 579–583, 1978.

[5] G.D. Borshukov, G. Bozdagi, Y. Altunbasak, and A.M. Tekalp. Motion Segmentation by Multi-stage Affine Classification. *IEEE Transactions on Image Processing*, 6(11):1591–1594, 1997.

[6] J. Cooper, S. Venkatesh, and L. Kitchen. Early Jump-Out Corner Detectors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(8):823–828, 1993.

[7] C. Coppini, M. Demi, R. Poli, and G. Valli. An Artificial Vision System for X-Ray Images of Human Coronary Trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(2):156–162, 1993.

[8] R. Deriche and O. Faugeras. 2-D Curve Matching Using High Curvature Points: Application to Stereo. In *Proceedings of 10th International Conference on Pattern Recognition*, pages 18–23, 1990.

[9] R. Deriche and G. Giraudon. A Computationnal Approach for Corner and Vertex Detection. *International Journal of Computer Vision*, 10(2):101–124, 1993.

[10] F. Deschênes and D. Ziou. Detection of Line Junctions and Line Terminations Using Curvilinear Features. Accepté pour publication dans Pattern Recognition Letters, 2000.

[11] A. El Zaart, D. Ziou, S. Wang, Q. Jiang, and G. B. Bénié. SAR Images Segmentation Using Mixture of Gamma Distribution. In *Proceedings of Vision Interface 99*, pages 125–130, Trois-Rivière, Canada, 1999.

[12] J. Ens and P. Lawrence. An Investigation of Methods for Determining Depth from Focus. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 15(2):97–108, 1993.

[13] B. Escalante and J.B. Martens. Noise Reduction in Computerized Tomography Images by means of Polynomial Transform. Technical Report no. 785, Institut for Perception Research, Eindhoven, The Netherlands, 1991.

[14] D. J. Fleet. Disparity from Local Weighted Phase-Correlation. In *IEEE International Conference on Systems, Man and Cybernetics*, volume 1, pages 48–54, 1994.

[15] D.J. Fleet and A.D. Jepson. Computation of Component Image Velocity from Local Phase Information. *International Journal of Computer Vision*, 5:77–104, 1990.

[16] M. F. Auclair Fortier, D. Ziou, C. Armenakis, and S. Wang. Automatic Update of Road Databases from Low Resolution Images. Technical report, Département de mathématiques et d'informatique, Université de Sherbrooke, 1999.

[17] M. Gokstrop. Computing Depth from Out-of-Focus Blur Using a Local Frequency Representation. In *Proceedings of the* 12^{th} *International Conference on Pattern Recognition*, pages 153–158, 1994.

[18] A. Horii. Depth from Defocusing. Tech. Report ISRN KTH/NA/P–92/16–SE, Royal Institute of Technology, Sweden, 1992.

[19] B.K.P. Horn. *Robot Vision*. MIT Press, Cambridge, MA, 1986.

[20] B.K.P. Horn and B.G. Schunck. Determining Optical Flow. *Artificial Intelligence*, 17:185–204, 1981.

[21] V. Kayargadde and J. B. Martens. Estimation of Edge Parameters and Image Blur Using Polynomial Transforms. *CVGIP: Graphical Models and Image Processing*, 56(6):442–461, 1994.

[22] L. Kitchen and A. Rosenfeld. Gray-level Corner Detection. *Pattern Recognition Letters I*, pages 95–102, 1982.

[23] V. Lacroix and M. Acheroy. Feature Extraction Using the Constrained Gradient. *ISPRS Journal of Photogrammetry and Remote Sensing*, 53:85–94, 1998.

[24] S.H. Lai, C.W. Fu, and S. Chang. A Generalized Depth Estimation Algorithm with a Single Image. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 14(4):405–411, 1992.

[25] J.B. Martens. The Hermite Transform - Applications. IEEE *Transactions on Acoustic Signal and Speech Processing*, 38(9):1607–1618, 1990.

[26] J.B. Martens. The Hermite Transform - Theory. IEEE *Transactions on Acoustic Signal and Speech Processing*, 38(9):1595–1606, 1990.

[27] F. Mokhatarian and R. Suomela. Robust Image Corner Detection Through Curvature Scale Space. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12):1376–1381, 1998.

[28] D.W. Murray and B.F. Buxton. Scene Segmentation from Visual Motion Using Global Optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(2):220–228, 1987.

[29] H.G. Musmann, M. Hotter, and J. Ostermann. Object-Oriented Analysis-Synthesis Coding of Moving Images. *Signal Processing: Image Communication 1*, pages 117–138, 1989.

[30] Z. Myles and N. V. Lobo. Recovering Affin Motion and Defocus Blur Simultaneously. In *Proceedings of IEEE, International Conference on Computer Vision and Pattern Recognition*, pages 756–763, San Francisco, US, 1996.

[31] H.H. Nagel. Displacement Vectors Derived from Second-Order Intensity Variations in Image Sequences. *Computer Graphics and Image Processing*, 21:85–117, 1983.

[32] S. Negahdaripour. Revised Definition of Optical Flow: Integration of Radiometric and Geometric Cues for Dynamic Scene Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(9):961–979, 1998.

[33] M. Nitzberg, D. Mumford, and T. Shiota. Filtering, Segmentation and Depth. *Lecture Notes in Computer Science*, 662, 1993.

[34] M. Ouali. Perception tridimensionnelle et reconstruction 3D. Mémoire de D.E.A., Centre de Robotique, Écoles des Mines de Paris, 1995.

[35] M. Ouali. Phase-Based Disparity and Blur Estimation. Rapport pré-doctoral, 1998.

[36] J.S. Park and J.H. Han. Estimating Optical Flow by Tracking Contours. *Pattern Recognition Letters*, 18(7):641–648, 1997.

[37] A. P. Pentland. Depth of Scene from Depth of Field. In *Proc. Image Understanding Workshop*, pages 253–259, 1982.

[38] A. P. Pentland, T. Darrell, M. Turk, and W. Huang. A Simple, Real-Time Range Camera. In *Proceedings of IEEE, International Conference on Computer Vision and Pattern Recognition*, pages 256–261, 1989.

[39] A.P. Pentland. A New Sense for Depth of Field. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 9(4):523–531, 1987.

[40] A.P. Pentland, S. Scherock, T. Darrell, and B. Girod. Single Range Camera Based on Focal Error. *J. Opt. Soc. Am. A*, 11(11):2925–2934, 1994.

[41] J. Perrin. Profondeur et binocularité: algorithmie, étude psychologique et intérêt pour l'ergonomie des interfaces stéréoscopique. Thèse de doctorat, Centre de Robo-

tique, Écoles des Mines de Paris, 1998.

[42] K. Prazdny. Detection of Binocular Disparities. *Biological Cybernetics*, 52:93–99, 1985.

[43] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling. *Numerical Recipies in C, The Art of Scienfitic Computing.* Cambridge University Press, 1988.

[44] R.J. Prokop and A.P. Reeves. A Survey of Moment-Based Techniques for Unoccluded Object Representation and Recognition. *CVGIP: Graphical Models and Image Processing*, 54:438–460, 1992.

[45] A.N. Rajagopalan and S. Chaudhuri. Space-Variant Approaches to Recovery of Depth from Defocused Images. *Computer Vision and Image Understanding*, 68:309–329, 1997.

[46] G. Sansone. *Orthogonal Functions.* Interscience Publishers, Inc., New York, 1959.

[47] J. Shen and W. Shen. Image Smoothing and Edge Detection by Hermite Integration. *Pattern Recognition*, 28(8):1159–1166, 1995.

[48] G.E. Sotak and K.L. Boyer. The Laplacian-of-Gaussian Kernel: A Formal Analysis and Design Procedure for Fast, Accurate Convolution and Full-Frame Output. *Computer Vision, Graphics and Image Processing*, 48:147–189, 1989.

[49] C. Steger. Extracting Curvilinear Structures: A Differential Geometric Approach. In *Proceedings of the Fourth European Conference on Computer Vision*, volume 1, pages 630–641, 1996.

[50] M. Subbarao and N. Gurumoorthy. Depth Recovery from Blurred Edges. In *Proceedings of IEEE, International Conference on Computer Vision and Pattern Recognition*, pages 498–503, 1988.

[51] M. Subbarao and G. Surya. Depth from Defocus: A Spatial Domain Approach. *The International Journal of Computer Vision*, 13(3):271–294, 1994.

[52] G. Sudhir, S. Banerjee, K. K. Biswas, and R. Bahl. Cooperative Integration of Stereopsis and Optic Flow Computation. *Journal of the Optical Society of America A*, 12(12):2564–2572, 1995.

[53] S. Tabbone. Detecting Junctions Using Properties of the Laplacian of Gaussian Detector. In *Proceedings of the 12th International Conference on Pattern Recognition*, pages 52–56, 1994.

[54] S. Tabbone and D. Ziou. On the Behavior of the Laplacian of Gaussian for Junction Models. In *Proceedings of the 2nd Annual Joint Conference on Information Sciences*, pages 304–307, USA, 1995.

[55] A.M. Tekalp. *Digital Video Processing.* Prentice Hall, Upper Saddle River, NJ, US, 1995.

[56] E. Trucco and A. Verri. *Introductory Techniques for 3-D Computer Vision.* Prentice Hall, Upper Saddle River, New Jersey, US, 1998.

[57] S. Uras, F. Girosi, A. Verri, and V. Torre. A Computational Approach to Motion Perception. *Biol. Cybern.*, 60:79–97, 1988.

[58] J.Y.A. Wang and E.H. Adelson. Representing Moving Images with Layers. *IEEE Transactions on Image Processing Special Issue: Image Sequence Compression*, 3(5):625–638, September 1994.

[59] R.G. White and R.A. Schowengerdt. Effect of Point-Spread Functions on Precision Edge Measurement. *J. Opt. Soc. Am. A*, 11:2593–2603, 1994.

[60] L. Wiskott. Segmention from Motion: Combining Gabor- and Mallat-Wavelets to Overcome Aperture and Correspondence Problem. In *Proceedings of the 7th International Conference on Computer Analysis of Images and Patterns*, pages 10–12, Kiel, Germany, 1997.

[61] Y. Xiong and S. A. Shafer. Depth from Focusing and Defocusing. In *Proceedings of IEEE, International Conference on Computer Vision and Pattern Recognition*,

pages 68–73, 1993.

[62] Y. Xiong and S. A. Shafer. Moment and Hypergeometric Filters for High Precision Computation of Focus, Stereo and Optical Flow. Tech. Report CMU-RI-TR-94-28, Carnegie Mellon University, 1994.

[63] D. Ziou. Passive Depth from Defocus Using a Spatial Domain Approach. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 799–804, Bombay, India, 1998.

[64] D. Ziou. Finding Lines in Grey-Level Image. Technical report, DMI, Université de Sherbrooke, Canada, J1K 2R1, 1999.

[65] D. Ziou and F. Deschenes. Depth from Defocus Estimation in Spatial Domain. Soumis à Computer Vision and Image Understanding, 1999.

[66] D. Ziou and S. Tabbone. Edge Detection Techniques - An Overview. *International Journal of Pattern Recognition and Image Analysis*, 8(4):1–40, 1998.